December 2020

Metric Learning via Linear Embeddings for Human Motion Recognition

ByoungDoo Kong

METRIC LEARNING VIA LINEAR EMBEDDINGS FOR HUMAN MOTION RECOGNITION

A Thesis Presented

by

BYOUNGDOO KONG

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE

September 2020

Electrical and Computer Engineering

METRIC LEARNING VIA LINEAR EMBEDDINGS FOR HUMAN MOTION RECOGNITION

A Thesis Presented

by

BYOUNGDOO KONG

Approved as to style and content by:

Marco F. Duarte, Chair

Mario Parente, Member

Hossein Pishro-Nik, Member

Christopher V. Hollot, Chair of the Faculty
Electrical and Computer Engineering

ABSTRACT

METRIC LEARNING VIA LINEAR EMBEDDINGS FOR HUMAN MOTION RECOGNITION

SEPTEMBER 2020

BYOUNGDOO KONG

B.S., DONGGUK UNIVERSITY

M.S.E.C.E., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Marco F. Duarte

We consider the application of Few-Shot Learning (FSL) and dimensionality reduction to the problem of human motion recognition (HMR). The structure of human motion has unique characteristics such as its dynamic and high-dimensional nature. Recent research on human motion recognition uses deep neural networks with multiple layers. Most importantly, large datasets will need to be collected to use such networks to analyze human motion. This process is both time-consuming and expensive since a large motion capture database must be collected and labeled. Despite significant progress having been made in human motion recognition, state-of-the-art algorithms still misclassify actions because of characteristics such as the difficulty in obtaining large-scale leveled human motion datasets. To address these limitations, we use metric-based FSL methods that use small-size data in conjunction with dimensionality reduction. We also propose a modified dimensionality reduction scheme based on the preservation of secants tailored to arbitrary useful distances, such as the geodesic distance learned by ISOMAP. We provide multiple experimental results that demonstrate improvements in human motion classification.

TABLE OF CONTENTS

Page

CHAPTER

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

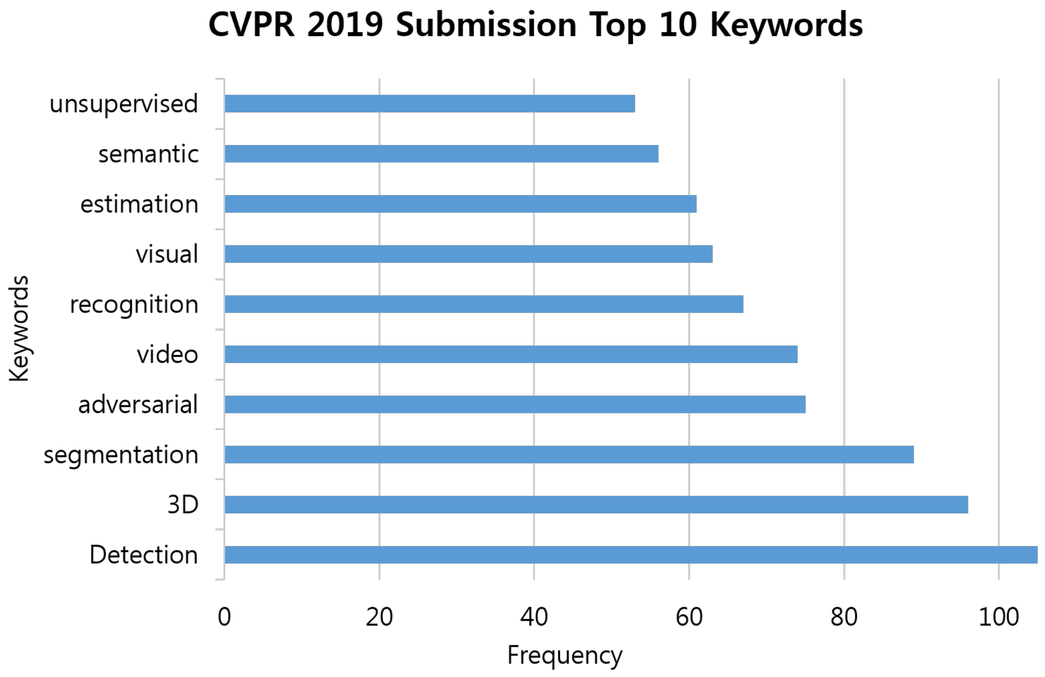


Figure 1.1: CVPR 2019 Submission Top 10 Keywords

Computer vision is the most important part of Artificial Intelligence (AI) [37]. Vision-based Human Motion Recognition (HMR) can be used in several fields such as visual surveillance, human-robot interaction, self-driving vehicle, and body language using human motion. Figure 1.1 shows the top 10 keywords among the Conference on Computer Vision and Pattern Recognition 2019 submissions [52]. Although we may use labeled datasets for human motion recognition, it is difficult to gather annotated data at large scale from real-world applications. Automatic labeling using Online

Training Architecture and movie scripts helps us to get motion annotation but still has a high level of label noise causing the negative effects on human motion recognition [37]. Even though the YouTube-8M and Sports-1M datasets used commonly in HMR provide very large sets of human motion videos, their annotations are obtained by a retrieval method and thus may be inaccurate. Therefore, we need to develop methods that learn novel objects from a few samples of the object as robots systems. Additionally, the high dimensionality of the datasets has negative effects on human motion modeling. These aspects provide multiple motivations for dimensionality reduction. The dimensionality reduction provides advantages such as visualization of the low dimensional structures in the data and reduces process time.

There are several popular methods to reduce the dimensionality. Principal Component Analysis (PCA) involves mapping the high dimensional data into the low dimensional subspace spanned by the influential eigenvectors of the covariance matrix. Although PCA discovers the average best fitting subspace in a least-squares sense, the average error metric can distort point cloud geometry [27]. We propose a linear embedding that used a geodesic distance or dynamic time warping (DTW) to preserve the geometric properties of the data. Human motion recognition requires the accurate motion alignment, which is computationally inefficient [18]. In contrast, we propose a simple yet the powerful method for human motion recognition, as shown by existing work such as NuMax [27] and manifold sub-sampling [39].

1.1 Purpose of the study

Although remarkable progress in HMR has been made, it is difficult to get higher performance using simple tasks due to the uniqueness of human action. It is not easy to gather large-scale annotated data for real applications. Furthermore, each video frame has a background that includes noise as well as human shapes. We need to use data conditioning such as motion and trajectory segmentation for action recognition.

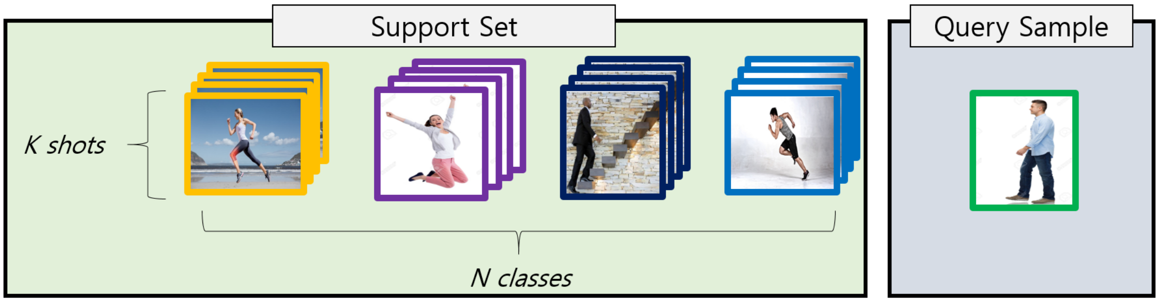


Figure 1.2: Overview of few-shot learning (4-way 4-shot).

This is because it is hard to accurately extract action features in videos. Moreover, it is important to choose a dimensionality reduction method that preserves the structure of data as much as possible. We propose the use of secant sets normalized with geodesic distances or DTW as the foundation of linear dimensionality reduction. We also evaluate the performance of dimensionality reduction techniques like PCA and ISOMAP for human motion recognition.

In this thesis, we propose the use of few shot learning based on meta learning for classification in HMR. Meta learning aims to learn a distribution for a new task from many past different tasks, instead of learning the representation of the classes [68]. The meta learning models are trained over a variety of learning tasks and optimized for the best performance on a distribution of tasks, including potentially unseen tasks. Each task is associated with a dataset D, containing both feature vectors and true labels. In other words, meta-learning, known as *learning to learn*, intends to make methods that can learn novel skills with a few training examples. There are three different approaches: 1) learn an efficient distance metric (metric-based); 2) apply recurrent network with external or internal memory (model-based); 3) optimize the model parameters for fast learning (optimization-based) [60]. Few-shot learning (FSL) is a field of meta-learning. We use **metric-based few-shot learning** [28] because FSL empirically work quite well [41, 51, 61] and we can use several methods to compare HMR features of datasets with different distances. The objective of FSL is

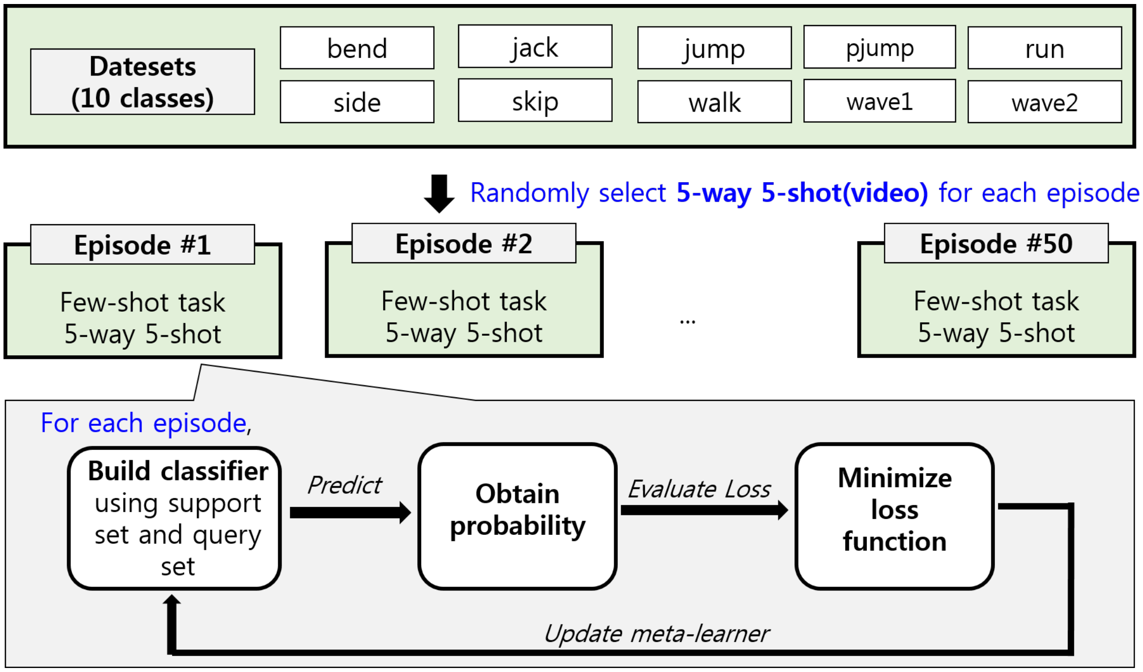


Figure 1.3: Process of FSL using meta(episode) learning.

to learn information from a small size of samples and to relieve the burden of getting annotated large-scale datasets [65].

Figure 1.2 shows the overview of few-shot learning under the setting of N-way K-shot classification. The support dataset consists of K samples per class for each of N classes of interest. Given a support set composed of N labels and, for each label K labeled images; a query set composed of Q query images, the task is to classify the query images among the N classes given the $N \times K$ images in the support set. When K is small (typicall $K < 10$), we talk about few-shot classification. Figure 1.3 explains the process of FSL through meta learning. The meta-learning adopt an episodic training strategy whereby each episode contains a meta-task [25]. Since we want our network to learn from a few data points, that is, we want to perform FSL, we train our network in the same way. Therefore, we use episodic training for each episode, we randomly sample a few data points from each class in our dataset and we call that a support set and train the network using only the support set, instead of the whole datset. Similarly, we randomly sample a point from the dataset as a query point and try to predic its class. So, in this way, our network is trained how to learn from a smaller set of data points. This is called FSL through meta-learning [44]. Most importantly, we combine dimensionality reduction and metric-based few-shot learning with feature extraction methods to discover the best solution for action classification.

1.2 Structure of the Thesis

Chapter 2 provides some approaches for feature extraction from human motion videos that will be used in the sequel: background subtraction [4], and Histogram of oriented gradients [20], and human optical flow [43]. We also discuss various dimensionality reduction methods such as NuMax, PCA and ISOMAP, which will be used in our numerical comparisons.

Chapter 3 introduces a modified dimensionality reduction method that aims to preserve a distance measure of interest to the application being considered, and shows its use in combination with several example recognition techniques. Our method is based on [27], which is reformulated to use a different distance, in particular, geodesic distance and dynamic time warping. The objective of our method is to preserve pairwise distances between human motion data points. After dimensionality reduction, we test several HMR classification approaches on the resulting data.

Chapters 4 and 5 present the results of various methodologies for HMR. The accuracy of classification depends on the data dimension and feature extraction methods. This section discusses future research to be applied to human motion.

CHAPTER 2

BACKGROUND

2.1 Action Representation

2.1.1 Space-Time Silhouettes

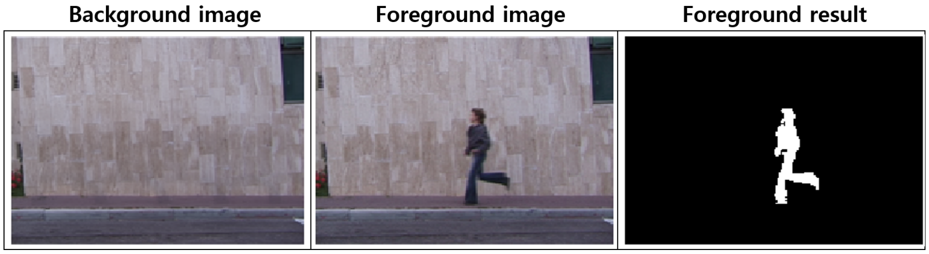


Figure 2.1: The process of background subtraction.

Videos of human motion consist of several frames. The objective of action representation is to extract a feature vector from motion videos [53]. The motion representation is considered as the temporal variation of human silhouettes. We use space-time silhouettes as basic inputs for human activity representation [11].

The human silhouettes can be obtained from background subtraction [4], which is the process of separating out foreground objects from the background in a sequence of video frames. Figure 2.1 shows the simple process of background subtraction. Background estimation uses the mean of the first few video frames to obtain an estimate of the scene, which we call the background frame. Foreground pixels are estimated as the deviation from the estimated background. We can then find moving objects using the difference between the current frame and the background image.

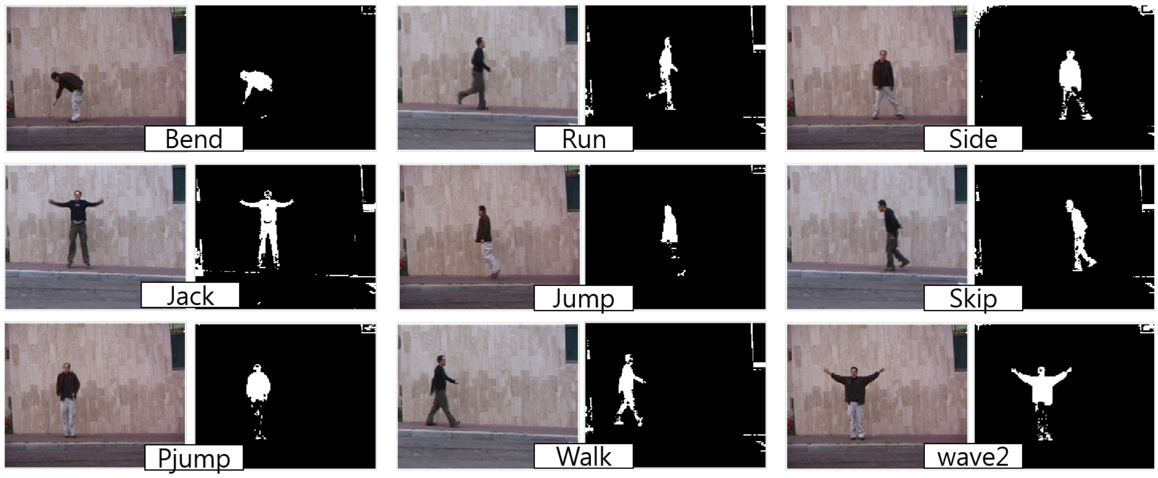


Figure 2.2: An example of the silhouette images (Weizmann datasets).

When we maintain a background frame, we can label a pixel (x, y) as belonging on the foreground if

$$|I(x,y,t) - B(x,y,t)| > \alpha \tag{2.1}$$

where α is the threshold, $I(x, y, t)$ is the image at time t and $B(x, y, t)$ is the background at time t. In contrast to some data classes such as run, walk, and jumping sideways, there are no objects in the first frames of data like bend, jump, and wave for the Weizmann datasets. Therefore, instead of using the first frames of each class, we experimentally compute the median of the previous n frames as the background image for image segmentation as the equation:

$$|I(x,y,t) - \text{median}\{I(x,y,t-i)\}| > \alpha \tag{2.2}$$

where $i \in \{0, .., n-1\}$. Figure 2.2 shows an example of the foreground images using the Weizmann datasets. The resulting images contain as much foreground as possible and do not distort the motion shape. For computational efficiency, the resolution of original data (180 x 144, 25 fps) is resized to 50% size and we use $\alpha = 0.5$.

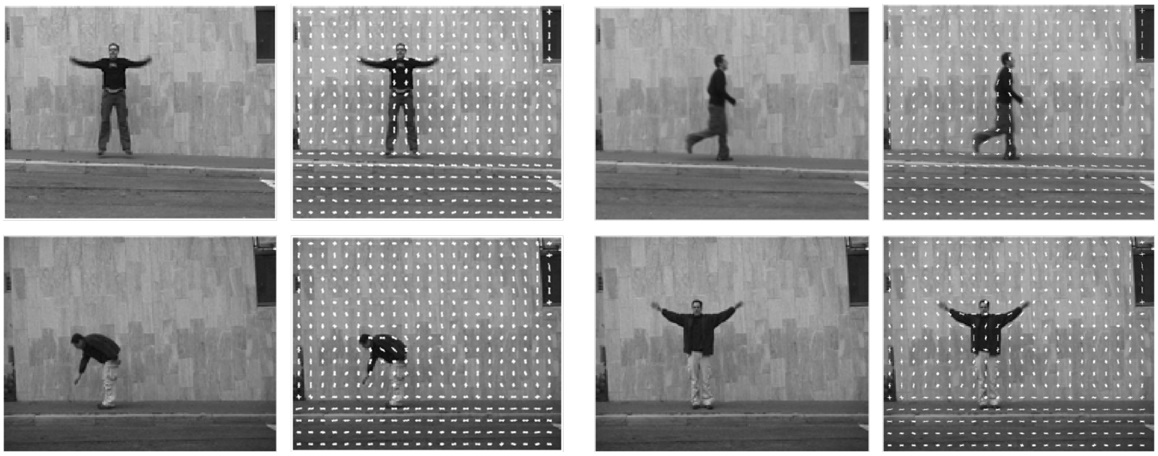


Figure 2.3: Example images and their histograms of gradients (HOGs).

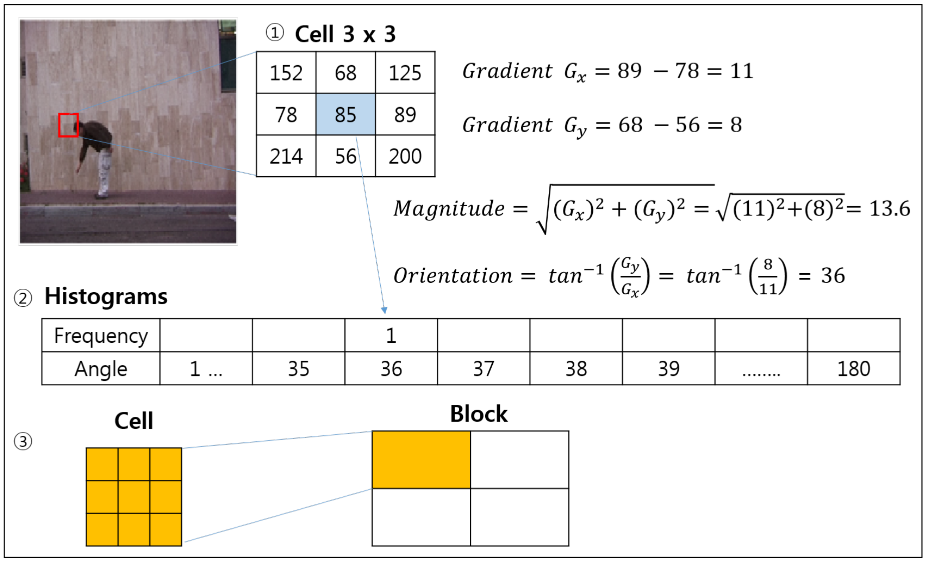
2.1.2 Histogram of Oriented Gradients

Histogram of Oriented Gradients (HOG) is one of the descriptors proposed for pedestrian detection [20]. HOG divides the target area into cells of the same size and calculates a histogram of the direction of edge pixels in each cell. These histogram bin values are connected in a row. HOG can be viewed as a directional histogram template of edge. HOG retains geometric information in block units and has some robust characteristics for local changes by using histograms of each block. HOG is a suitable feature extraction for identifying objects with simple contour because it uses silhouette information of objects. Figure 2.3 shows the example images of HOG.

The process of HOG descriptor is as follows:

1. Divide each image into 8×8 cells and compute the gradients of direction x, y and magnitude, and orientation. The gradients are the change in size and direction.

2. Compute histograms with gradients and orientation. A histogram is a graph of the frequency of each interval for a variable. The orientation and angle value of each pixel are used to make a frequency table shown in Figure 2.4. The frequency table can be converted to the histogram with angle values (x-axis) and the frequency (y-axis). The concept of bin also can be used in this process



152	68	125
78	85	89
214	56	200

$Gradient\ G_x = 89 - 78 = 11$

$Gradient\ G_y = 68 - 56 = 8$

$Magnitude = \sqrt{(G_x)^2 + (G_y)^2} = \sqrt{(11)^2 + (8)^2} = 13.6$

$Orientation = tan^{-1}\left(\frac{G_y}{G_x}\right) = tan^{-1}\left(\frac{8}{11}\right) = 36$

Frequency			1					
Angle	1 ...	35	36	37	38	39	……..	180

Figure 2.4: The algorithm of Histogram of oriented gradients (HOG) Descriptor

as each section of a histogram. We divide the image into 8×8 cells and make the histograms, we obtain a 9×1 matrix for each cell.

The magnitude is the ℓ_2-norm of the gradient vector:

$$\text{Magnitude} = \sqrt{(G_x)^2 + (G_y)^2} \tag{2.3}$$

where G_x, G_y is the x, y-gradient. The orientation is arctangent of the ratio between the partial derivatives on two directions:

$$\text{Orientation} = \tan^{-1}\left(\frac{G_x}{G_y}\right) \tag{2.4}$$

3. Normalize the histograms. Since the lighting of each cell is different, gradients normalization is applied to mitigate the effect of lighting variations. For example, 8×8 cells are collected into a 16×16 block, and we normalize the matrix using the Euclidean norm.

2.1.3 Human Optical Flow

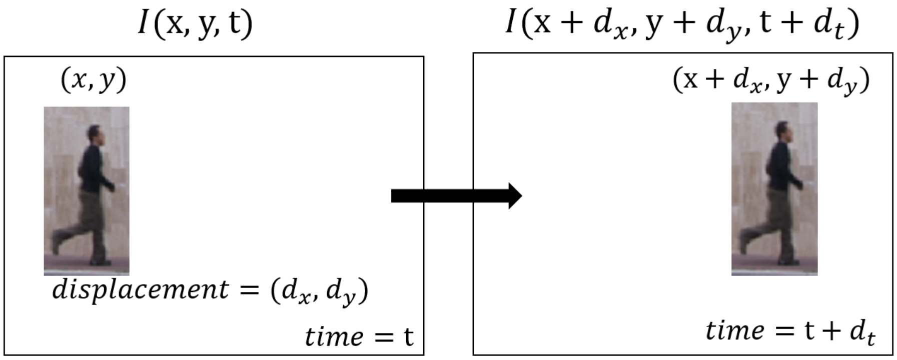

$I(\mathrm{x},\mathrm{y},\mathrm{t})$
$I(\mathrm{x}+d_x,\mathrm{y}+d_y,\mathrm{t}+d_t)$
(x,y)
$(\mathrm{x}+d_x,\mathrm{y}+d_y)$
$displacement = (d_x, d_y)$
$time = \mathrm{t}$
$time = \mathrm{t}+d_t$

Figure 2.5: The concept of the optical flow.

Optical flow detects movement by comparing the difference between two consecutive images [43]. It is assumed that the brightness of the object is preserved among images. If there is a noticeable change of brightness between the two adjacent images, the velocity of the movement is calculated. In other words, optical flow is caused by a change of brightness, whether an object is stationary or not. It also assumes time persistence, i.e., that the movement between frames is small. The change can be seen as the differential of brightness values over time. Space-adjacent objects are more likely to be the same object. Figure 2.5 shows the concept of human optical flow. Optical flow is the movement of human motions between continuous frames of sequences due to the relative movement between human shape and the camera. The optical flow can be represented as follows:

$$I(x, y, t) = I(x + d_x, y + d_y, t + d_t) \tag{2.5}$$

The image intensity $I(x, y, t)$ is a current frame and $I(x + d_x, y + d_y, t + d_t)$ is the next frame after moving by t. An assumption of pixel intensities is needed to apply the optical flow: there is just a little or no difference in brightness constancy between consecutive images.

By applying Taylor series approximation, equation (2.5) is changed to

$$I(x + d_x, y + d_y, t + d_t) = I(x, y, t) + \frac{\partial I}{\partial x} d_x + \frac{\partial I}{\partial y} d_y + \frac{\partial I}{\partial t} d_t + ... \tag{2.6}$$

Plugging (2.5) in (2.6) and removing common terms,

$$\frac{\partial I}{\partial x} d_x + \frac{\partial I}{\partial y} d_y + \frac{\partial I}{\partial t} d_t = 0. \tag{2.7}$$

The optical flow equation is derived by dividing by d_t:

$$\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} = 0, \tag{2.8}$$

where $\partial I/\partial x$, $\partial I/\partial y$ and $\partial I/\partial t$ are the image gradients of the horizontal axis, the vertical axis, and time, and $u = d_x/d_t$ and $v = d_y/d_t$ are the velocity or optical flow of $I(x+y+t)$. Lucas-Kanade method [57] and Farneback optical flow [29] are proposed to solve the optical flow equation because there is one equation against two unknown variables: u and v. The Farneback method [29] is a dense optical flow that uses the flow vectors of the entire frame, while the Lucas-Kanade method is one of the sparse optical flow use few pixels of the edges or corners of human shapes. In this paper, we use Farneback optical flow to improve the classification accuracy despite being more computationally expensive than the Lucas-Kanade method. The Farneback optical flow calculates the optical flow vector for all pixels of each image. The Farneback optical flow estimates the windows of the frames by quadratic polynomials through polynomial expansion transform [22]. The polynomial expansion approximates some neighborhood of each pixel with a polynomial [29]. The Farneback optical flow uses quadratic polynomials expressed in a local coordinate system as below:

$$f(x) \sim x^T A x + b^T x + c \tag{2.9}$$

Where A is a symmetric matrix, b is a vector and is c a scalar [29]. We assume that the image $f_1(x)$ is taken at time t and $f_2(x)$ at time $t+d_t$. The polynomial coefficients of the previous frame are connected to the ones from the second frame by applying a displacement d (2.10 $\sim$ 2.12).

$$f_2(x) = f_1(x-d) = (x-d)^T A_1 (x-d) + b_1^T (x-d) + c_1 \tag{2.10}$$

$$= x^T A_1 x + (b_1 - 2A_1 d)^T x + d^T A_1 d - b_1^T d + c_1 \tag{2.11}$$

$$= x^T A_2 x + b_2^T x + c_2. \tag{2.12}$$

By assuming the brightness of two continuous images, we can get

$$A_2 = A_1, \quad b_2 = b_1 - 2A_1 d, \quad c_2 = d^T A_1 d - b_1^T d + c_1. \tag{2.13}$$

Here d is can be solved using b_2 of (2.13) as

$$d = -\frac{1}{2} A_1^{-1} (b_2 - b_1). \tag{2.14}$$

The $-\frac{1}{2}(b_2 - b_1)$ term of equation (2.14) can be modified to Δb.

$$\Delta b = -\frac{1}{2}(b_2(x) - b_1(x)). \tag{2.15}$$

We also write

$$A(x) = \frac{A_1(x) + A_2(x)}{2}. \tag{2.16}$$

Therefore, the equation (2.14) can be changed to

$$A(x)d(x) = \Delta b(x). \tag{2.17}$$

By assuming that all pixels in a window I follow the same model, d can be computed using A_1, b_1 and c_1 from f_1 in each pixel in I. As mentioned in [22], we make the

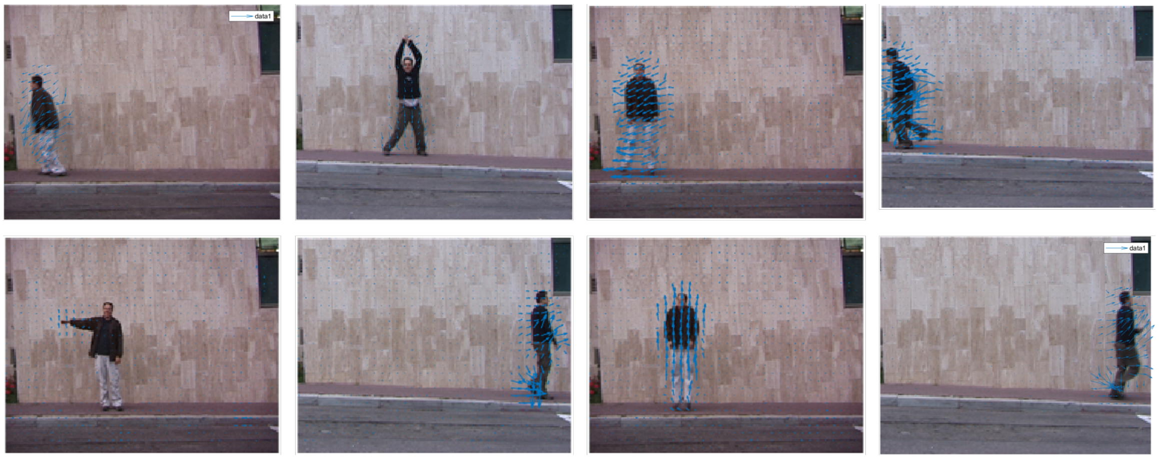


Figure 2.6: Example images of the optical flow

assumption that the displacement field is slowly varying, so that we can integrate information over a neighborhood of each pixel. Therefore, we try to find $d(x)$ satisfying the equation (2.17) as well as possible over a neighborhood I of x, or more formally minimizing:

$$e(X) = \sum_{\Delta x \in \mathbf{I}} w(\Delta x) || A(x + \Delta x) d(x) - \Delta b(x + \Delta x) ||^2, \tag{2.18}$$

where we let $w(\Delta x)$ be a Gaussian weight function for the points in the neighborhood. So, we care more about the center of the neighborhood than the edges [9]. The $w(\Delta x)$ is used to reflect the influence degree of each point in the neighborhood area. The closer each pixel is to the target pixel in the neighborhood area, the greater the value of the Gaussian weighting function is. The displacement $d(x)$ of the target point during the corresponding time is determined by minimizing the error function $e(X)$, from which the velocity of the pixel is deduced as shown the equation (2.18). Figure 2.6 shows example images of human optical flow for the Weizmann datasets.

2.2 Dimensionality Reduction

To get a compact description and efficient computation, we consider efficient methods of dimensionality reduction. Dimensionality reduction provides an embedding onto a low dimensional Euclidean space. In other words, the dimensionality reduction enforces a distance metric to match the Euclidean distance in the embedded target space. There are several popular methods for dimensionality reduction e.g., isometric mapping (ISOMAP), local linear embedding (LLE) and Linear Discriminant Analysis (LDA).

2.2.1 Principal Component Analysis (PCA)

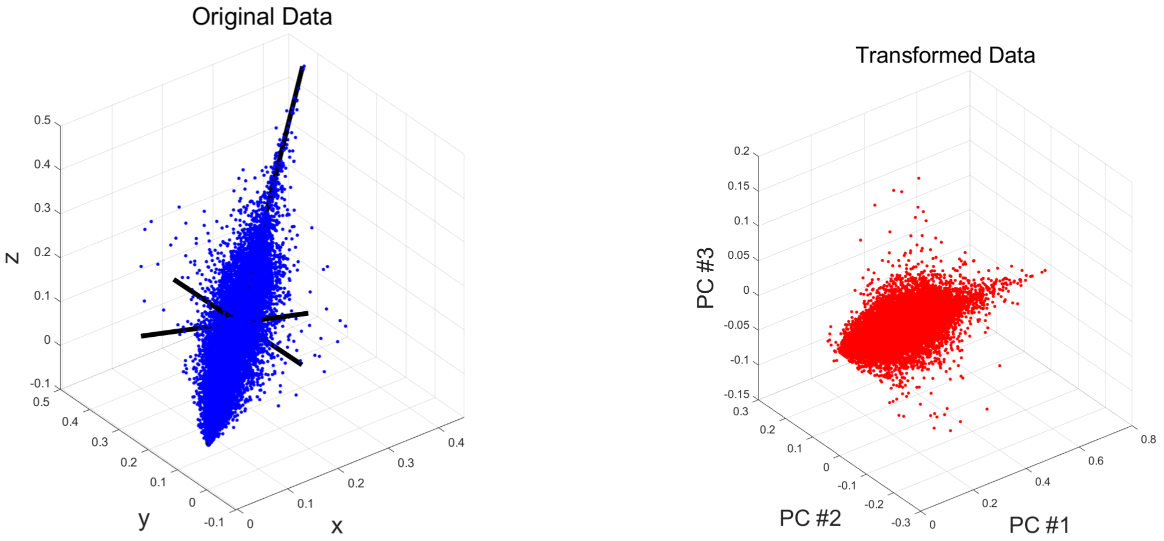
Principal component analysis (PCA) is a technique for reducing data in high-dimensional spaces to low-dimensional space. The dimension of the PCA embedding is less than or equal to the dimension of the original sample. The principal component analysis is a projection of the original data onto the directions in which the variance is greatest. The variance is another measure of the spread of the datasets. Figure 2.7 shows that PCA finds the axes of the principal component of data. PCA calculates the mean and covariance of the data. The covariance measures the direction of the linear relationship between variables.

$$\mathrm{cov}(x,y) = E[(x-m_x)(y-m_y)] = E[xy] - m_x m_y, \tag{2.19}$$

where m_x and m_y are the average of x and y, $E[\cdot]$ is the expected value. If there are two dimensional data (x_1, y_1), (x_2, y_2), $....(x_n, y_n)$, the covariance matrix is calculated as follows:

$$\mathbf{C} = \begin{pmatrix} \mathrm{cov}(x,x) & \mathrm{cov}(x,y) \\ \mathrm{cov}(x,y) & \mathrm{cov}(y,y) \end{pmatrix} \tag{2.20}$$

In this case, PCA finds the eigenvalue and eigenvector that best describes the datasets. The eigenvector with the highest eigenvalue is the principal component of the data.



(a) (b)

Figure 2.7: Example of PCA, "Run" of Weizmann datasets. (a) Original Data. (b) Transformed Data.

Let A be a square a matrix, v a vector and λ a scalar that satisfies $Av = \lambda v$, then λ is a eigenvalue associated with eigenvector v of A. The eigenvalues of A are roots of the characteristic equation:

$$\det(A - \lambda I) = 0 \tag{2.21}$$

By sorting the eigenvectors by decreasing eigenvalues and choosing k eigenvectors, a $d \times k$ dimensional eigenvector matrix W is constructed using eigenvectors with the highest eigenvalues. PCA explains the existing datasets or predicts new datasets using eigenvalue and eigenvector.

2.2.2 ISOMAP

We use Isometric Mapping (ISOMAP) as a non-linear embedding to compare with the performance of linear embeddings. Linear embeddings such as PCA reduce the dimensions based on Euclidean distances, whereas ISOMAP uses the geodesic distance approach among the multivariate data points. ISOMAP is a non-linear dimensionality reduction method widely used for manifold learning and based on a spectral theory which tries to preserve the geodesic distances in the lower dimension [58]. The manifold is a topological space that on small scale resembles the Euclidean space of a specific dimension. ISOMAP seeks lower levels of embedding to maintain geodetic distance between all points. Geodesic distance is the distance between two stations along an elliptical plane.

The first step in ISOMAP is to measure which points are located close to each other on the manifold and connect all points using a k-nearest neighbor (KNN) graph. In the second step, ISOMAP calculates the geodesic distance between all pairs of points using Dijkstra or Floyd's algorithm with the aforementioned graph. Finally, ISOMAP applies multidimensional scaling (MDS) to the matrix of graph distances. MDS finds an embedding space $\mathcal{Y}$, when given a matrix D containing the pairwise geodesic distances, that minimize the differences between the Euclidean distances $\|y_i - y_j\|$ in the embedded space and distance matrix entries to preserve all pairwise geodesic distances.

2.2.3 NuMax

Nuclear norm minimization with Max-norm constraints (NuMax) [27] is a near-isometric linear dimensionality reduction technique. Given a dataset $X \subset R^N$ that contains Q points, NuMax discover a linear embedding $P : R^N \rightarrow R^M$, $M \ll N$, that satisfies the restricted isometry property (RIP) on X with parameter $\delta > 0$, referred to as the isometry constant [27, 54]. The RIP is defined as follows:

$$(1-\delta)||x-x'||_2^2 \;\leq\; ||Px-Px'||_2^2 \;\leq\; (1+\delta)||x-x'||_2^2. \tag{2.22}$$

Both x and x' are points from the dataset. NuMax use the secant set of $S = \binom{Q}{2}$ unit vectors $\mathcal{S}(X) = \{v_1, v_2, ..., v_s\}$. The normalized secant set of x is proposed by Whitney Reduction Network (WRN) [15] as follows:

$$\mathcal{S}(X) = \left\{ \frac{x - x'}{||x - x'||_2}, \quad x, x' \in X, \quad x \neq x' \right\}. \tag{2.23}$$

NuMax computes a projection matrix $\Psi \in R^{M \times N}$ with as few rows as possible that satisfies the RIP on $\mathcal{S}(X)$ [27]. Let $S^{N \times N}$ be the set of symmetric $N \times N$ matrices. NuMax defines $P = \Psi^T \Psi \in S^{N \times N}$ and rank$(P) = M$. NuMax uses the constraints $|||\Psi v_i||_2^2 - 1| = |v_i^T P v_i - 1| < \delta$ for every secant v_i in $\mathcal{S}(X)$. Let 1_s denote the $\mathcal{S}$-dimensional all-ones vector, and let A denote the linear operator that maps a symmetric matrix X to the S dimensional vector $A : X \to \{v_i^T X v_i\}_{i=1}^S$. NuMax finds the matrix P as the optimization problem.

$$\begin{aligned} &\text{minimize} \quad \text{rank}(P) \\ &\text{subject to} \quad ||A(P) - 1_S||_\infty \leq \delta, \;\; P \geq 0. \end{aligned} \tag{2.24}$$

NuMax computes a nuclear-norm relaxation of (2.25) as proposed in [24, 45] since rank minimization is a non-convex problem:

$$\begin{aligned} &\text{minimize} \quad ||P||_* \\ &\text{subject to} \quad ||A(P) - 1_S||_\infty \leq \delta, \;\; P \geq 0 \end{aligned} \tag{2.25}$$

The nuclear norm of P is the same as its trace because P is a positive semidefinite symmetric matrix. When the solution $P^* = U \Lambda U^T$ of equation (2.26) is computed, rank(P^*) decides the value of the dimensionality M of the linear embedding M. The linear embedding Ψ can be computed using a simple matrix square root:

$$\Psi = \Lambda^{1/2}{}_M U_M^T \tag{2.26}$$

where $\Lambda_M = \text{diag}\{\lambda_1, \lambda_2, ..., \lambda_M\}$ denotes the M leading non-zero eigenvalues of P^* and U_M denotes the set of corresponding eigenvectors. NuMax returns a low-rank matrix $\Psi \in R^{M \times N}$ that satisfies the RIP on the secant set $\mathcal{S}(X)$ with isometry constant δ. These concepts of NuMax are used in Chapter 3 to make a novel linear dimensionality reduction.

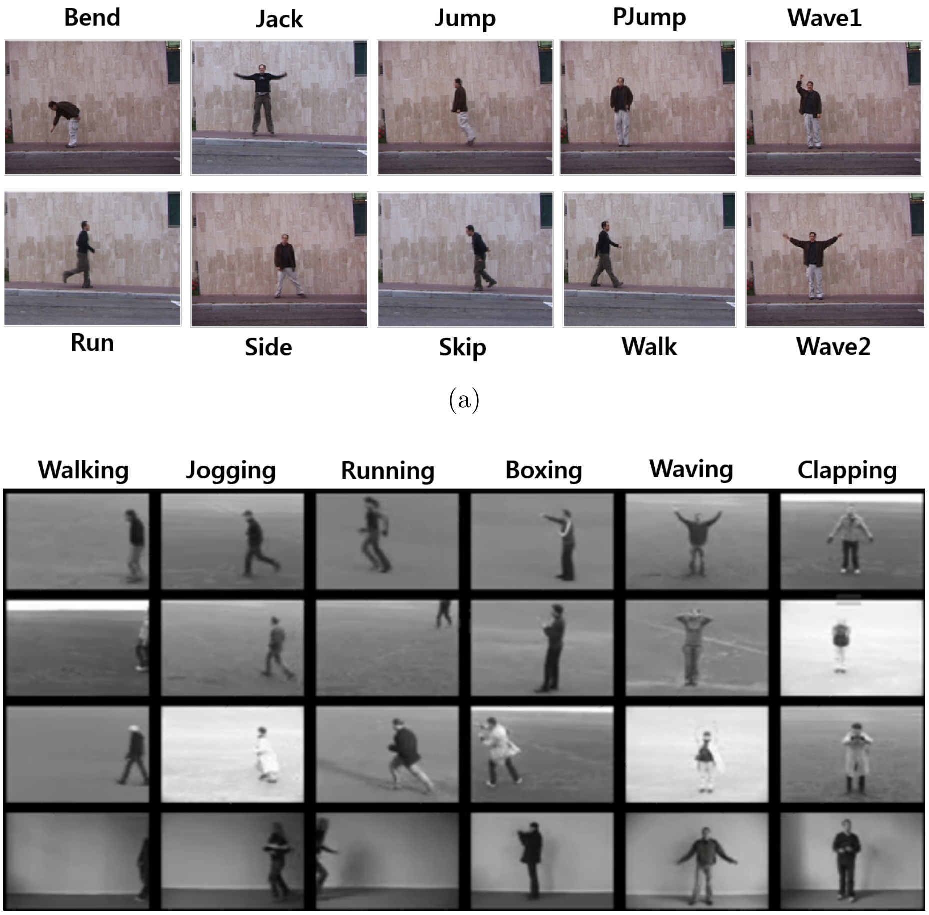
2.3 Related Work in Computer Vision

Human motion recognition is a rapidly growing branch of computer vision. In recent years, due to developments in artificial intelligence and convolutional neural networks (CNN), the accuracy of human motion recognition is mostly improving. For example, L. Wang et al. [64] use deep learning architectures for human motion recognition in videos; and S. Ji et al. [32] provide convolutional neural networks (CNN) based on 3D convolutions to extract features from temporal and spatial dimensions. H. Wang et al. [62] propose an approach to classify videos through dense trajectories. The motion trajectory is an efficient method to model long-period motion. There are several techniques such as histogram of oriented gradients (HOG), a histogram of optical flow (HOF) and motion boundary histograms (MBH) descriptors to discover the trajectories [59, 62]. Kalouris et al. [34] combine data augmentation such as rotation, scale and jitter, and transfer learning to improve the performance of prediction, and performance and robustness. Huseyin et al. [18] propose metric learning using a triplet architecture and recurrent neural network (RNN) to control sizes of embedding. Kalouris et al. [37] summarize the pros and cons of motion recognition approaches. Shallow methods for human motion recognition such as space-time and sequential techniques are easy to use and can achieve high performance. However, each technique needs extra annotations on the data. On the other hand, methods like hybrid and multi-stream methods are straightforward to implement using present convolution networks. But these methods are difficult to fine-tune.

Although state-of-the-art techniques show high performance in results, they mostly need to gather a large amount of annotated data. Negative effects on incorrect labeling should also be considered. Since the use of many layers in convolutional neural networks (CNN) is related to the pervasive use of nonlinearities, it is useful to extract features from the data. The larger and more complex the network is, the more time it takes to learn and test. Therefore, some of these methods are computationally expensive. To solve these limited conditions, FSL techniques are being developed. FSL uses only a few data points in its training datasets. FSL uses a support data set for training and query data for testing. These tasks of FSL are called N-way K-shot. For example, 2-way 5-shot approach use two categories like dog and cat and five images per category. Similar to an example of deep learning that uses millions of images of cats and dogs, increasing the value of shots K improves classification accuracy. FSL is simply the idea of using only a few labeled samples. FSL learns how to make the data of the same classes be closer while keeping data from different classes separated. Although some FSL techniques use graph neural networks to maximize the classification performance, we use FSL based on distance learning such as prototypical networks and matching networks.

2.4 Datasets

Our experiments for this thesis use two datasets. The Weizmann dataset [12, 26] is consists of short video segments and has ten different actions: running, jumping in place, jumping forward, bending, waving with one hand, jumping jack, jumping sideways, jumping on one leg, walking, and waving with two hands from 9 people. The resolution is 180 × 144 pixels, and we randomly select 6 people for training and 3 people for testing. The datasets contain 90 videos. The KTH dataset [48] are consists of 6 different types of activities: boxing, handclapping, handwaving, jogging, running and walking performed by 25 subjects in 4 different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. The total number of videos is 600 (25×4×6) for each combination of 25 subjects, 6 actions



(a)

(b)

Figure 2.8: Human motion datasets. (a) Weizmann datasets. (b) KTH datasets.

and 4 scenarios. Each video contains about 4 sub sequences used as a sequence in our experiments. Table 2.1 represents the size of two datasets before and after being applied feature extraction.

Dataset	*Methods*	*Original dataset*			*After feature extraction*	
		Height	*Width*	*frames*	*Feature vector*	*Frame length*
Weizmann	Background Subtraction	144	180	$28 \sim 50$	25920	19
	HOG				12852	21
	Optical flow				25920	25
KTH	Background Subtraction	120	160	$25 \sim 200$	19200	19
	HOG				9576	25
	Optical flow				19200	50

Table 2.1: Summary of datasets and extracted features.

CHAPTER 3

MULTI-COMBINATION METHODS FOR HUMAN MOTION CLASSIFICATION

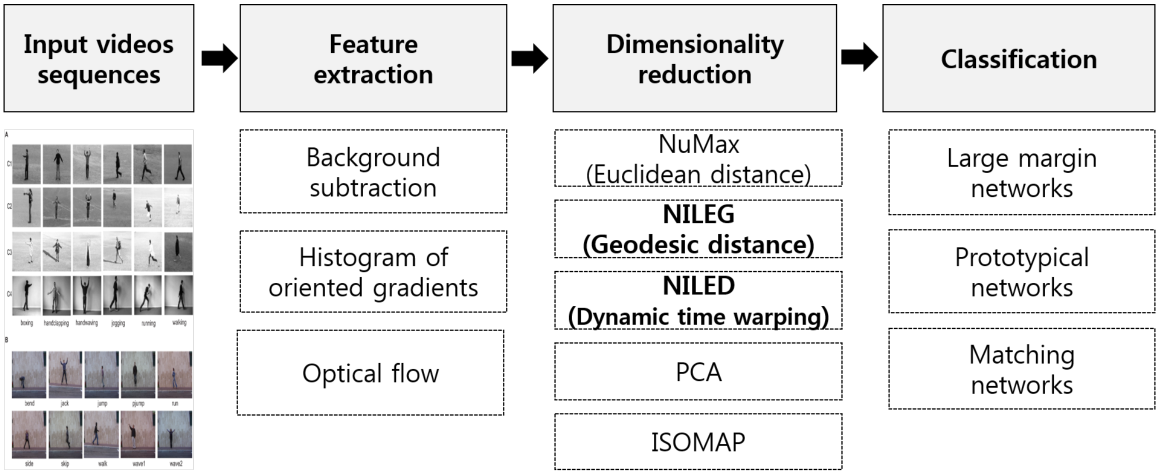


Figure 3.1: The flowchart of the proposed framework of human motion recognition

We propose an integrated framework, as shown in Figure 3.1, for the task of human motion recognition. The proposed framework consists of three major parts: **feature extraction**, **dimensionality reduction** and **motion classification** in an embedded space. In particular, we propose a customized dimensionality reduction method to preserve the intrinsic structure of the motion feature vectors that is useful in human motion recognition. The objective of dimensionality reduction is to reduce the negative effects of the curse of dimensionality [38], which causes lower performance of motion classification. We use various dimensionality reduction techniques after feature extraction for learning algorithms. Finally, we apply metric-based FSL to improve the classification accuracy for human motion recognition.

3.1 Linear Embeddings from Optimization

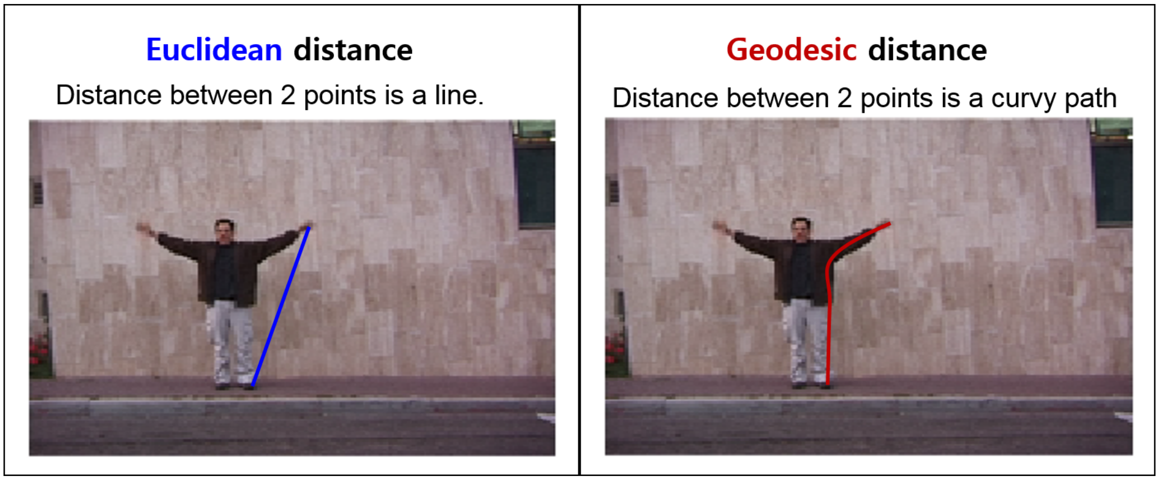
In this paper, we modify NuMax [27], which is a linear dimensionality reduction as mentioned in Section 2.2.3, to use a different distance when defining the secant set. We generally apply the concept of NuMax [27] as a linear embedding, but we design a variant that uses a custom distance measure to be leveraged in the application of interest. We use two distances: geodesic distance and dynamic time warping (DTW) distance. First of all, we employ geodesic distance, which is a generalization of a straight line to curved surfaces, to preserve the human motion vector as much as possible. With neighbors known, the shortest path geodesic distance (nonlinear distance) between each pair of samples in the dataset are determined [5]. In contrast to Euclidean distance, the geodesic distance depends on the manifold where the points lie. Firstly, we pick and connect k nearest neighbor points based on the input space distance:

$$d(x_i, x_{i+1}) = ||x_i - x_{i+1}|| \tag{3.1}$$

These neighborhood relations are represented as a weighted graph G over the data points. We build a graph using a k-NN approach. We then apply Dijkstra's algorithm with the nearest neighbor graph G to discover the shortest-path distances for all pairs of data points. We apply the shortest-path distances $D(x_i, x_{i+1})$ to normalize the corresponding secants $s = x_{i+1} - x_i$. The geodesic distance with respect to a data set D, a distance $d(u, v)$ and a neighborhood k are defined as follows:

$$D(a, b) = \min_p \sum_i d(p_i, p_{i+1}) \tag{3.2}$$

where p is a sequence of points of length $l \geq 2$ with $p_1 = a$, $p_l = b$, $p_i \in D$ $\forall i \in \{2, ..., l-1\}$ and (p_i, p_{i+1}) and k-nearest-neighbors [8]. Figure 3.2 illustrates the concept of Euclidean distance and geodesic distance. We calculate geodesic distance using the same procedure as ISOMAP [36]. Inspired from NuMax [27] and SLRNILE [54],


Euclidean distance
Distance between 2 points is a line.
Geodesic distance
Distance between 2 points is a curvy path

(a) (b)

Figure 3.2: Two different types of distances. (a) Euclidean distance. (b) Geodesic distance.

we find the low-rank linear mapping that satisfies the restricted isometric property (RIP) condition on the secant set. Unlike in [27,54], we use a different secant set based on geodesic distance. The **secant set based on geodesic distance** is defined as follows :

$$S(x_i, x_{i+1}) = \left\{ \frac{x_i - x_{i+1}}{D(x_i, x_{i+1})}, \quad x_i, x_{i+1} \in X, \quad x_i \neq x_{i+1} \right\} \tag{3.3}$$

where x_i and x_{i+1} are the pairwise pixels of frame. The secant vector can be regarded as a normalized difference vector between two different pixels for human motion datasets. The objective of our method is to find a linear embedding matrix $P : R^N \rightarrow R^M, M \ll N$. Inspired from NuMax [27], we find a projection matrix $\Phi \in R^{M \times N}$ that satisfies the RIP on our secant set $S(x_i, x_{i+1})$ based on geodesic distance.

NuMax has to satisfy the restricted isometry property (RIP) on the secant set (3.2). RIP is defined as follows:

$$(1-\delta)||x_1 - x_2||_2 \leq ||\Phi x_1 - \Phi x_2|| \leq (1+\delta)||x_1 - x_2||_2. \tag{3.4}$$

where δ is referred to as the isometry constant [27]. When δ approaches zero, both ends of the inequality become closer to each other. In this case, the embedding matrix Φ is the best matrix which allows only for minor corruption to the geometric information of the datasets. The embedding matrix Φ does not alter the secant vectors much in terms of their magnitude. Therefore, it is important to calculate the embedding matrix Φ with RIP properties. We estimate the variation of the isometry constant δ with the number of measurements M. Through (3.2) and (3.4), the RIP condition of linear embedding matrix P are changed as follows:

$$(1-\delta)||S_l||_2 \;\leq\; ||\Phi S_l|| \;\leq\; (1+\delta)||S_l||_2. \tag{3.5}$$

where S_l is a secant in the set $\mathcal{S}(X)$. Consider a linear transform $A : R^{N\times N} \rightarrow R^K$ as $A(\Phi^T\Phi) : \Phi^T\Phi \rightarrow \{v_l^T\Phi^T\Phi v_l\}_{l=1}^K$ where $K = |S(\Phi^T\Phi)|$ for every secant v_i in $\mathcal{S}(X)$. The output of $A(\Phi^T\Phi)$ is a K dimensional vector with the lth entry being $v_l^T\Phi^T\Phi v_l$. As mentioned in NuMax [27] and SLRNILE [54], we link the number of rows of Φ with the rank of $\Phi^T\Phi$ and find an optimal linear mapping using rank minimization together with the isometry constraint:

$$\begin{aligned} &\text{minimize} \quad \text{rank}(P) \\ &\text{subject to} \quad ||A(P) - 1_K||_\infty \leq \delta,\;\; P \geq 0. \end{aligned} \tag{3.6}$$

where 1_K is a K dimensional all ones vector and the ℓ_∞ norm $||.||_\infty$ ensures that the worst case of distortion for any secant vector satisfies the RIP condition with isometry constant δ [6]. Since affine rank minimization, which aims at finding a matrix of minimum rank that satisfies a given system of linear equality constraints, is an NP-hard problem to solve, we use convex relaxation to obtain a trace-norm minimization [19]. A common trick to solve the rank-minimization problem approximately is to replace the rank function with the nuclear norm of the matrix. The nuclear norm is the sum of the singular values of the matrix [45]. Given a symmetric matrix $X \in R^{N \times N}$, the nuclear norm of X, denoted by $||X||_*$, is equal to the sum of its singular values, or equivalently, the ℓ_1-norm of σ:

$$||X||_* = \sum_{i=1}^{r} \sigma_i(X). \tag{3.7}$$

The nuclear norm is a convex function, can be optimized efficiently, and is the best convex approximation of the rank function [27, 45]. Thus, the problem of rank minimization is solved using the nuclear norm, the convex relaxation of rank penalty. In other words, the relaxation of the rank minimization that is more computational amenable is the following nuclear norm minimization:

$$\begin{aligned} &\text{minimize} \quad ||P||_* \\ &\text{subject to} \quad ||A(P) - 1_K||_\infty \le \delta, \;\; P \ge 0. \end{aligned} \tag{3.8}$$

As mentioned in NuMax [27], we use the Alternation Direction Method of Multipliers (ADMM) to solve the optimization (3.8). ADMM is an approximate algorithm that divides the complex equation into sub-problems easier for optimization. A convex problem with a linear constraint can be solved by using an auxiliary variable. We rewrite the equation (3.8) by applying the auxiliary variables $L \in S^{N \times N}$ and $q \in R^K$ to get the optimization problem:

$$\min_{P,L,q} ||P||_* \quad \text{subject to} \;\; ||q - 1_K||_\infty \le \delta, \;\; P = L, \;\; q = A(L), \;\; P \ge 0 \tag{3.9}$$

Next, we relax the linear constraints and form an augmented Lagrangian of (3.9) and update all variables by solving sub-problems with the ADMM [13]:

$$\begin{aligned} &\min_{P,L,q} ||P||_* + \frac{\beta_1}{2}||P - L - \triangle||_F^2 + \frac{\beta_2}{2}||A(L) - q - \omega||_2^2 \\ &\qquad \text{subject to} \;\; ||q - 1_K||_\infty \le \delta, \;\; P = L, \;\; q = A(L), \;\; P \ge 0. \end{aligned} \tag{3.10}$$

where the symmetric matrix $\triangle \in R^{N \times N}$ and vector $\omega \in R^{K}$ show the scaled Lagrange multipliers with a defined parameter $\beta_1, \beta_2 > 0$. We then optimize P, Φ, q, $\triangle$ and ω with iterative procedures and updates each variable at the stopping iteration $n+1$ get the low-dimensional embedding Y of human motion data while keeping the others fixed. When setting the other variables fixed, we gets a new estimate of q_{k+1} by solving the optimization program:

$$q_{k+1} \longleftarrow \arg\min_{q} \frac{\beta_2}{2}||A(L_k) - \omega_k - q||_2^2, \text{ subject to } ||q - 1_K||_\infty \leq \delta \tag{3.11}$$

We then denote $z = A(L^k) - \omega^k - 1_K$ to obtain the closed form solution $q_{k+1} = 1_K + \text{sign}(z) \cdot \min(|z|, \delta)$, where the operators $\text{sign}(\cdot)$ and $\min(\cdot)$ are applied componentwise. When setting the other variables fixed, P_{k+1} is updated by solving the following objective function:

$$P_{k+1} \longleftarrow \arg\min_{P} ||P||_* + \frac{\beta_1}{2}||P - L_k - \triangle_k||_2^F, \text{ subject to } P \geq 0. \tag{3.12}$$

We denotes $P' = L_k + \triangle_k$ and perform the eigendecomposition $P' = V\Sigma V^T$, where $\Sigma = \text{diag}(\sigma)$. The optimum P_{k+1} then can be shown as

$$P_{k+1} = VD_\alpha(\Sigma)V^T, D_\alpha(\Sigma) = \text{diag}(\{(\sigma_i - \alpha)_+\}), \tag{3.13}$$

where $\alpha = 1/\beta_1$ and t_+ represents the positive part of t. Isolating the terms that involve L, we get a new estimate L_{k+1} as the solution of the unconstrained optimization problem

$$L_{k+1} \longleftarrow \arg\min_{L} \frac{\beta_1}{2}||P_k - L - \triangle_j||_2^F + \frac{\beta_2}{2}||A(L) - q_{k+1} - \omega||_2^2 \tag{3.14}$$

The least-squares problem can be updated by solving the linear system

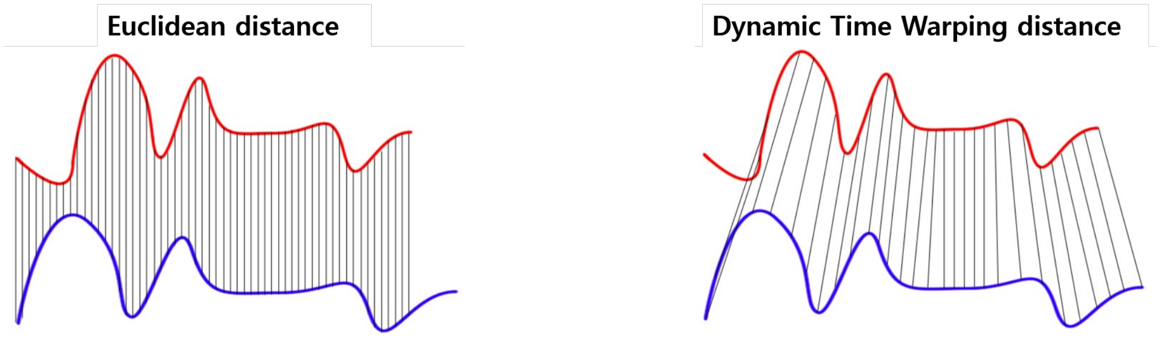
$$\beta_1(P_k - L - \triangle_j) = \beta_2 A^*(A(L) - q_{k+1} - \omega_k), \tag{3.15}$$

where A^* is the adjoint of A. Finally, having the q_{k+1}, P_{k+1}, L_{k+1}, $\triangle_{k+1}$ and ω_{k+1} are iteratively updated using a gradient ascent scheme with the step size of μ on the Lagrange multipliers as follows:

$$\triangle_{k+1} \longleftarrow \triangle_k - \mu(P_k - L_k), \;\; \omega_{k+1} \longleftarrow \omega_k - \mu(A(L_k) - q_k) \tag{3.16}$$

This process is repeated until the number of iterations exceeds the predefined maximum. We dub our method NILEG, an abbreviation for **Near-Isometric Linear Embeddings using Geodesic distance**.

We also use dynamic time warping (DTW) distance, which is defines the discrepancy between two time series. In time series analysis, DTW is one of the algorithms for measuring similarity between two temporal sequences, which may vary in speed. DTW is a method to compare two time series which may be different in length, DTW operates by trying to find the optimal alignment between two time series by means of dynamic programming [30]. DTW provides an approximate similarity measurement while allowing for matching partially identical sequences. Suppose we have two different arrays red and blue with different length as shown Figure 3.3. Although these two series follow the same pattern (Figure 3.3(a)), the blue curve is longer than the red. Using DTW allows us to match the troughs and peaks with the same pattern (Figure 3.3(b)). DTW is a method that calculates an optimal match between two given sequences with certain restriction and rules [67]. For example, every index from the first sequence must be matched with one or more indices from the other sequence. Also the first index from the first sequence must be matched with the first index from the other sequence. In the first sequence, the mapping of indices to indexes of other sequence increases monotonically. More specifically, given two time series $x_1, ..., x_n$ and $y_1, ..., y_m$, the DTW distance $D(i, j)$ is calculated as the equation (3.17).



(a)

(b)

Figure 3.3: Dynamic Time Warping (DTW) process in time series. (a) Euclidean distance between two arrays. (b) DTW between two arrays.

$$D(i,j) = \left\{ \begin{array}{c} D(i,j-1) \\ D(i-1,j) \\ D(i-1,j-1) \end{array} \right\} + d(x_i, y_i). \tag{3.17}$$

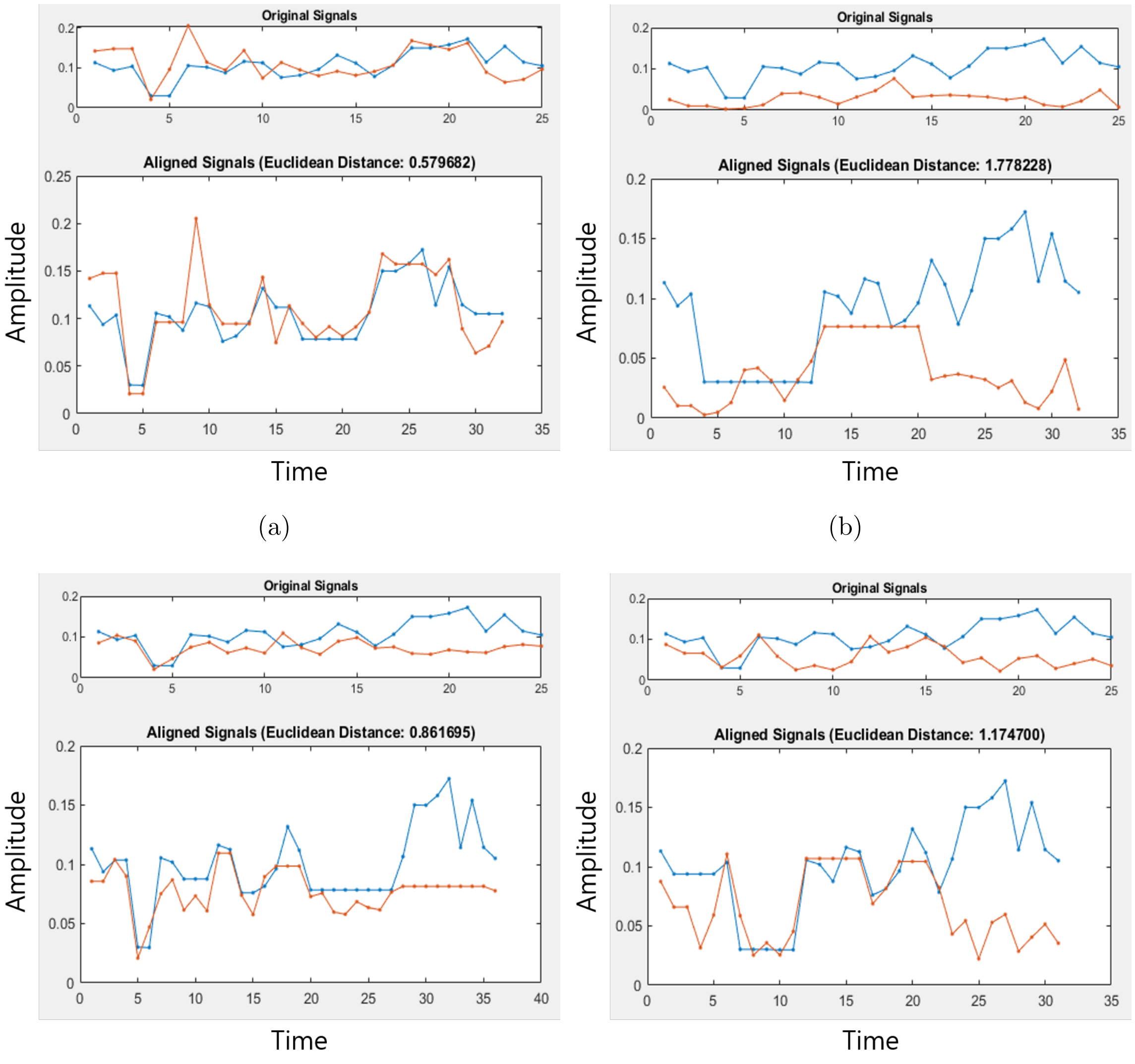
where $d(\cdot,\cdot)$ is the local distance function specific to application. Equation (3.17) means that the DTW distance between two arrays with length i and j equals the distance between the tails and the minimum of cost in arrays with length $i-1$, j, i, $j-1$, and $i-1$, $j-1$. Figure 3.4 and Table 3.1 shows the examples of DTW for data points from the Weizmann dataset. We vectorize each frame and calculate DTW between two consecutive frames for HMR. DTW distance between same classes is smaller than distances between different classes. We use DTW distance to make secant set and then apply the process of linear Embeddings from optimization. we call this method as NILED, an abbreviation for **Near-Isometric Linear Embeddings using DTW**. The secants set based on DTW distance is defined as follows :

$$\mathcal{S}(x) = \{d_1, d_2, ..., d_l\} \tag{3.18}$$

where d_l is made by reindexing of $D(i,j)$ of the equation (3.17) for consecutive frames.

two sequences	boxing vs boxing	walking vs boxing	running vs boxing	jogging vs boxing	handwaving vs boxing	handclapping vs boxing
DTW distance	**0.5796**	1.7782	0.8616	1.1747	1.3516	0.969

Table 3.1: The examples of DTW distance for the Weizmann datasets.



(a)

(b)

(c)

(d)

Figure 3.4: DTW distances of two sequences. (a) boxing vs boxing dataset. (b) walking vs boxing. (c) running vs boxing. (d) jogging vs boxing.

3.2 Classification

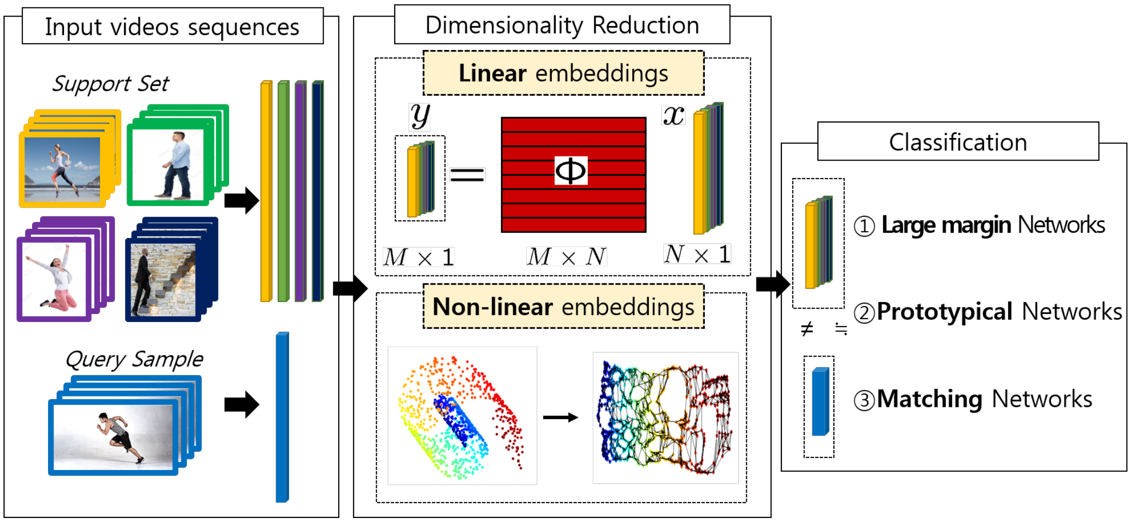


Figure 3.5: The pipeline of the proposed methods

In this chapter, we compare the classification performance of several combinations according to methods of feature extraction and dimensionality reduction by varying the embedded dimensions M of the low dimensional embeddings. We use classifiers such as large margin networks, prototypical networks and matching networks as shown in Figure 3.5. The objective of these methods is to gather the same classes to improve classification accuracy.

3.2.1 Large Margin Networks

Large Margin Nearest Neighbors (LMNN) is an algorithm that uses Mahalanobis distance [66]. Mahalanobis distance is a distance measurement method that takes into account the standard deviation of variables as well as the correlation between variables. This metric can solve the problem that Euclidean distance does not take into account the correlation of datasets, and is also scale-invariant. The Mahalanobis distance is defined between two vectors x and y as follows [66]:

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1}(\vec{x}-\vec{y})} \tag{3.19}$$

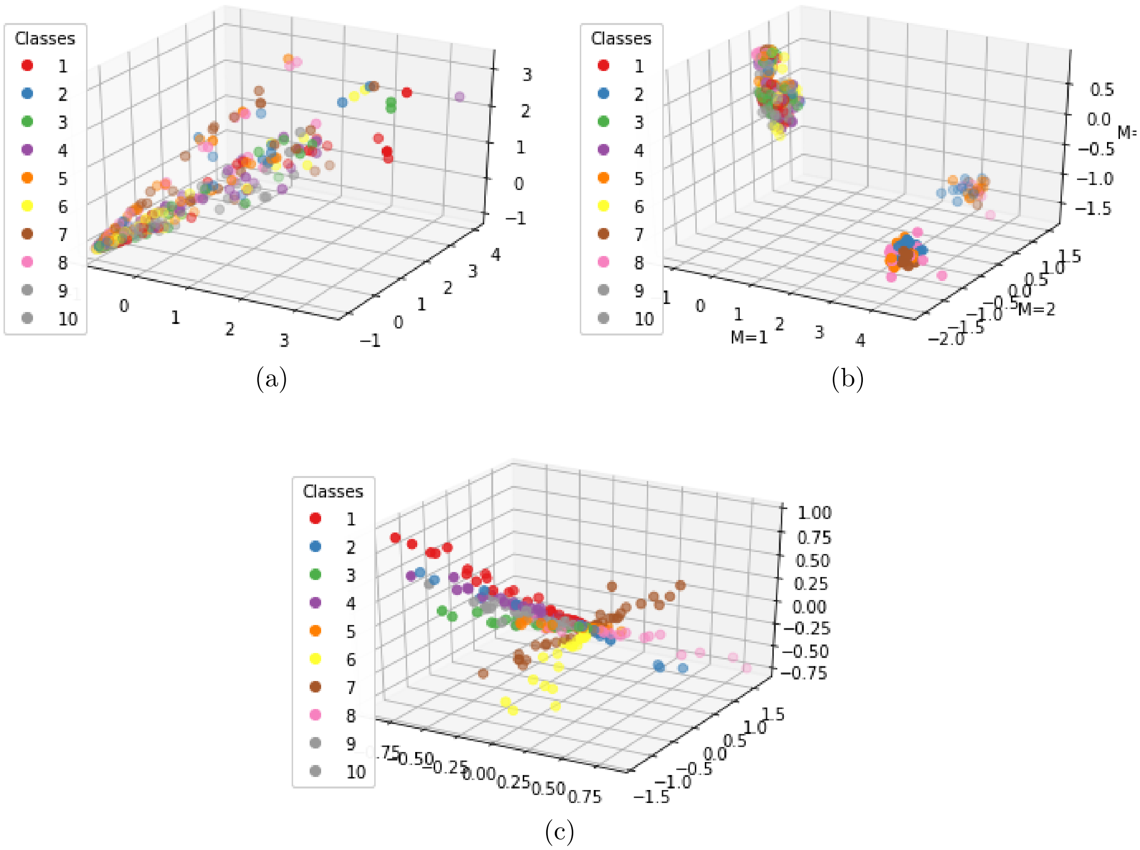
where S is the class covariance matrix. However, it is difficult to calculate the covariance matrix of the random datasets whose actual distribution is unknown. LMNN is a method for learning the covariance matrix S [7]. As mentioned in [66], we get a family of metrics over a vector space X by computing Euclidean distances after performing a linear transformation $\vec{x'} = L\vec{x}$. The linear transformation L is chosen to maximize the variance of the projected inputs, subject to the constraint that L defines a projection matrix. The large margin networks tries to learn a matrix $M = L^T L$. that maximizes the distances between examples with different labels and minimizes the distances between nearby examples with the same label. The Mahalanobis distance can be represented as

$$d(\vec{x_i},\vec{x_j}) = ||L(\vec{x_i}-\vec{x_j})||^2 \tag{3.20}$$

where the linear transformation in the equation (3.17) is parameterized by the matrix L. The cost function of LMNN can be written as

$$\varepsilon(L) = \sum_{ij}\eta_{ij}||L(\vec{x_i}-\vec{x_j})||^2 \ + c\sum_{ijl}\eta_{ij}(1-y_{il})h[1+||L(\vec{x_i}-\vec{x_j})||^2 - ||L(\vec{x_i}-\vec{x_l})||^2], \tag{3.21}$$

where η_{ij} is a binary variable that indicates whether $\vec{x_j}$ is a target neighbor of $\vec{x_i}$, and y_{il} is a binary variable that represents whether label y_i and y_l are equal to each other, $h(x)$ is the so-called hinge function $h(x) = \max(0,x)$, c is a weight that controls a trade-off between the pull and push terms. The first term pulls distances between inputs and target neighbors, while the second term pushes distances between datasets that have different labels. In other words, the first term minimizes the distances for the same labels between target neighbors. The second term pushes the distances of target neighbor x_i and x_j as far away as possible for l which have different labels. We refer to the differently labeled inputs in the training set that invade this perimeter



(a)

(b)

(c)

Figure 3.6: Effect of Large Margin Networks (x-axis:M= 1, y-axis:M= 2, Z-axis:M= 3)(Classes 1: bend, 2: jack, 3: jump, 4: pjump, 5: run, 6: side, 7: skip, 8: walk, 9: wave1, 10: wave2). (a) Before Margin Networks (HOG+NILEG). (b) After Margin Networks (HOG+LMNN). (c) After Margin Networks (HOG+NILEG+LMNN).

as impostors. The impostors are differently labeled neighbors defined by a simple inequality [66]. The second term is generated by the equation (3.19). This is achieved by penalizing distances to impostors $\vec{x}_l$ that are less than one unit further away than target neighbors $\vec{x}_j$ and therefore pushing them out of the local neighborhood of $\vec{x}_i$. For an input $\vec{x}_i$ with label y_i and target neighbor $\vec{x}_j$, an impostor is any input $\vec{x}_l$ with label $\vec{y}_l \neq \vec{y}_i$ as follows:

$$d(\vec{x}_i, \vec{x}_j) + 1 \leq d(\vec{x}_i, \vec{x}_l). \tag{3.22}$$

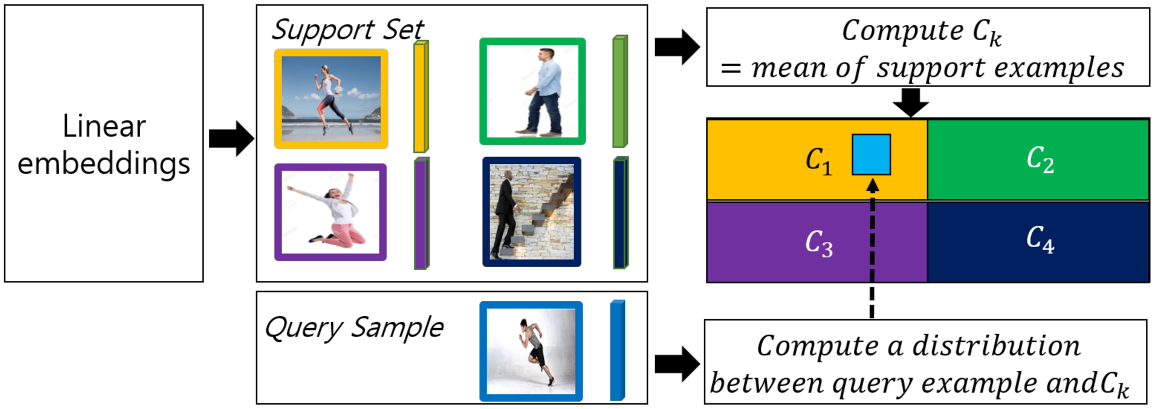

Linear embeddings
Support Set
Query Sample
Compute C_k = mean of support examples
C_1
C_2
C_3
C_4
Compute a distribution between query example andC_k

Figure 3.7: Prototypical Networks for human motion recognition

In contrast to previous approaches, we use NILEG and NILED as dimensionality reduction to solve the overfitting issue as mentioned in [66]. We also use $\vec{x_i}$ as support set to apply metric-based few-shot learning. Basically, our method use 5-way 5-shots(videos) as metric-based FSL. Figure 3.6 shows the large margin networks classification results for the Weizmann datasets. Finally, $K = 3$ nearest neighbor method is used as classifier [35]. Figure 3.6(a) shows the classification performance when we use histogram of oriented gradients as feature extraction and NILEG for dimensionality reduction. After being applied large margin networks, each class of datasets is gathered separately according to labels as shown in Figure 3.6(c). We also tested in cases of combinations of HOG+LMNN without any dimensionality reduction as shown in Figure 3.6(b). When we use the combination of HOG+NILEG+LMNN, the classes of human motion datasets are gathered well. We repeat our experiments for various combinations of feature extraction, dimensionality reduction and classification to prove the performance of NILEG and NILED. We set the number of neighbors $K = 5$ to make a secant set based on geodesic distance in NILEG.

3.2.2 Prototypical Networks

We use prototypical networks that learn a metric space in which few-shot classification can be performed by computing Euclidean distances to prototype representations of each class [41, 51]. Figure 3.7 shows the performance of prototypical networks for human motion recognition. We use a support set of N labeled examples $S = \{(x_1, y_1), ..., (x_N, y_N)\}$ and $y_i \in \{1, ..., K\}$ is the corresponding label. S_k denotes the set of examples labeled with class k. First of all, the prototypical networks compute the mean of the support vectors to obtain a class prototype C_k for each class:

$$C_k = \frac{1}{|S_k|} \sum_{(x_i, y_i) \in S_k} f(x_i) \tag{3.23}$$

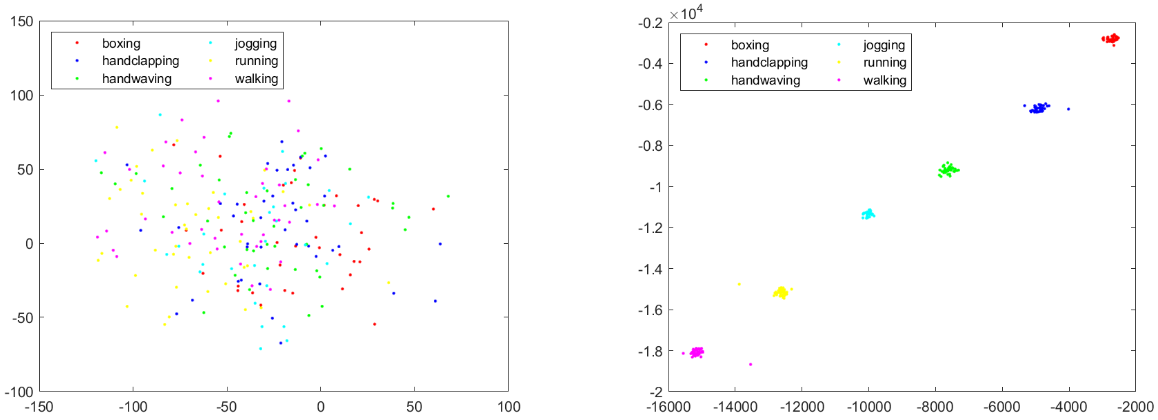
where S_k is the support set belonging to class k and f is the embedding function. We use our dimensionality reduction methods like NILEG, NuMax and ISOMAP as the embedding function. The embedding vectors f are calculated from feature extraction and dimensionality reduction, and we calculate Euclidean distances between the feature vector of the query image and the class prototype. Prototypical Networks make a distribution over classes for a query point x based on a softmax mapping distances to the prototypes in the embedding space [51]:

$$P(y = k|x) = \frac{\exp(-d(f(x), c_k))}{\sum_{k'} \exp(-d(f(x), c_{k'}))} \tag{3.24}$$

where d is the squared Euclidean distance. Its goal is to maximize the cross-entropy with the prototypes-based probability expression. The following loss function is minimized using the negative log-probability of the true class k via stochastic gradient descent (SGD):

$$J = -\text{log}P(y = k|x) \tag{3.25}$$

Finally, we compute the loss J for a randomly generated training episode. Our approach uses human motion vectors composed from a succession of frames as inputs and applies the prototypical networks after linear embeddings to make simpler without convolution operation unlike existing works related to image classification.



(a) (b)

Figure 3.8: Effect of Prototypical Networks (x-axis: $M = 1$, y-axis: $M = 2$). (a) Before Prototypical Networks(Background Subtraction + NILEG). (b) After Prototypical Networks(Background Subtraction + NILEG + Prototypical Networks).

Algorithm 1 Prototypical Networks algorithm

```
Step 1 → Human motion datasets $D$, comprising $\{(x_1, y_1), ..., (x_n, y_n)\}$ where $x$ is the feature and $y$ is the class label.
Step 2 → Randomly select sample 5 videos of data points per each action class for our datasets, $D$, and prepare our support set, $S$.
Step 3 → Similarly select 5 videos of data points and prepare the query set, $Q$.
Step 4 → Compute the mean (=prototype $C_k$) of each class.
Step 5 → Calculate the Euclidean distance, $d$, between query set and each prototype.
Step 6 → Predict the probability, $P(y = k|x)$, of the class of the query set by using softmax over distance $d$.
Step 7 →Compute the loss function, $J(\theta)$, as the negative log probability and try to minimize the loss using stochastic gradient descent.
```

Prototypical networks pull the distance between class prototype C_K and a query sample for the same class while keeping prototypes from other classes separate. In other words, the process of the prototypical networks is as follows:

3.2.3 Matching Networks

Matching networks compute an attention matrix from a cosine similarity between the support set and the query set [61], as shown in Figure 3.9. Matching networks multiply the attention matrix and one-hot encoder to calculate a probability for each

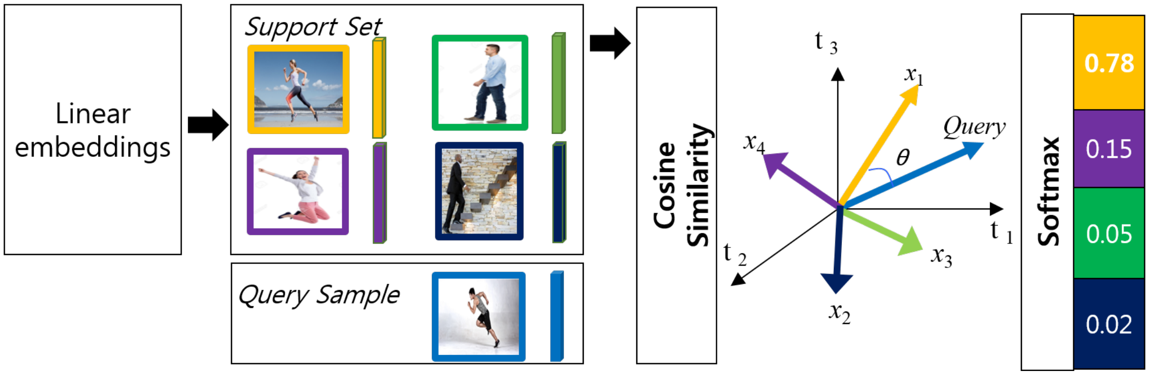


Figure 3.9: Matching Networks for human motion recognition

class. Next, softmax is applied to make an attention map. Cosine similarity calculates the cosine of the angle between two vectors to find the similarities in subspace. The larger the cosine value, the smaller the angle and the greater the match between vectors. Given two vectors, A and B, the cosine similarity, $\cos(\theta)$, is represented as

$$\text{similarity} = \text{cosine(A,B)} = \frac{A \cdot B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{3.26}$$

In other words, matching networks are a method for learning a classifier C_S with a small data set of support set $S = \{x_i, y_i\}_{i=1}^{k}$ (k-shot). Given the test samples X, the classifier defines a probability distribution for the output label y. Similar to other metric-based models, the output of the classifier is defined as the weighted label sum over the support samples using weights from the attention kernel $a(x, x_i)$. The value of the attention kernel should be proportional to similarity level between the images:

$$C_S(x) = P(y|x, S) = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i, \text{ where } S = \{(x_i, y_i)\}_{i=1}^{k}, \tag{3.27}$$

where $P(y|x, S)$, the predicted probability, is the weighted sum of the labels y_i, of the support set; x_i is the input of the support set; $\hat{x}$ is the query input; and a is a kernel function that calculates the similarity between x_i and x. The attention kernel is changed according to f and g, which are the embedding functions for the query and support set samples respectively. The attention weight $a(\hat{x}, x_i)$ between two data points depends on the the cosine similarity (cosine(·)) between two embedding vectors:

$$a(\hat{x}, x_i) = \text{softmax}(\text{cosine}(f(\hat{x}), g(x_i))) = \frac{\exp(\text{cosine}(f(\hat{x}), g(x_i))}{\sum_{j=1}^{k} \exp(\text{cosine}(f(\hat{x}), g(x_j))} \tag{3.28}$$

Vinyals et al. [61] use short-term memory (LSTM) recurrent neural network to embed the support and test datasets. In contrast, we use the embeddings obtained by our dimensionality reduction methods as the input to the matching networks. Therefore, we do not use the LSTM network for embeddings of the support set and query set. Our method is simpler than the related work, which uses Convolutional Neural Network (CNN) to obtain feature vectors [17].

CHAPTER 4

EXPERIMENTAL RESULTS

In this section, we show experimental results using three different classification methods on two datasets of human motion. NILEG is compared against other dimensionality reduction techniques including NuMax, PCA and ISOMAP. Table 4.1 provides a summary of the classification accuracy on the Weizmann dataset. The classification results represent that the combination of HOG, NILEG, and Large margin networks outperforms other combination methods in embedded dimension M (classification accuracy: 99.47%). NILEG preserves the nearest neighborhood of the datasets and therefore gets better classification accuracy of the human motion datasets. We prove that the combination of feature extraction, dimensionality reduction, and metric-based few-shot learning performs the improvement of the classification accuracy through multiple evaluations.

Analogous evaluation tests were taken concerning the KTH datasets. Although the combination of the best classification accuracy is different between the Weizmann datasets and the KTH datasets, NILEG obtains the higher classification accuracies in

Methods		*NILEG*		*NILED*		*NuMax*		*PCA*		*ISOMAP*	
		M	*Accuracy*	*M*	*Accuracy*	*M*	*Accuracy*	*M*	*Accuracy*	*M*	*Accuracy*
Large margin networks	Background	220	99.42	90	97.86	190	95.24	190	92.26	160	88.1
	HOG	**190**	**99.47**	130	99.31	220	99.21	110	97.22	210	91.67
	Opticalflow	220	85.29	150	93.75	240	58.33	60	64.29	90	59.92
Prototypical networks	Background	390	94.53	270	93.59	390	99.17	The Best combination : HOG + NILEG + Large margin networks			
	HOG	150	98.62	300	94.72	390	89.63				
	Opticalflow	240	92.92	300	94.58	120	98.33				
Matching networks	Background	60	56.08	90	63.42	60	65.04				
	HOG	360	66.46	60	44.58	390	66.08				
	Opticalflow	720	77.86	420	83.40	120	32.33				

Table 4.1: Summary of evaluation results for the Weizmann datasets [%].

Methods		*NILEG*		*NILED*		*NuMax*		*PCA*		*ISOMAP*	
		M	*Accuracy*	*M*	*Accuracy*	*M*	*Accuracy*	*M*	*Accuracy*	*M*	*Accuracy*
Large margin networks	Background	630	96.93	270	94.64	690	99.18	390	99.66	420	93.45
	HOG	420	99.25	510	96.28	510	96.84	150	94.54	300	89.61
	Opticalflow	510	94.7	510	97.95	450	74.54	150	28.15	420	25.53
Prototypical networks	**Background**	**180**	**99.85**	270	99.43	240	96.97	The Best combination : HOG + NILEG + Large margin networks			
	HOG	210	99.44	120	94.95	270	99.44				
	Opticalflow	180	97.65	30	98.02	30	89.56				
Matching networks	Background	600	80.76	540	82.45	720	68.28				
	HOG	510	91.84	720	98.96	480	94.45				
	Opticalflow	690	97.3	90	97.80	720	84.32				

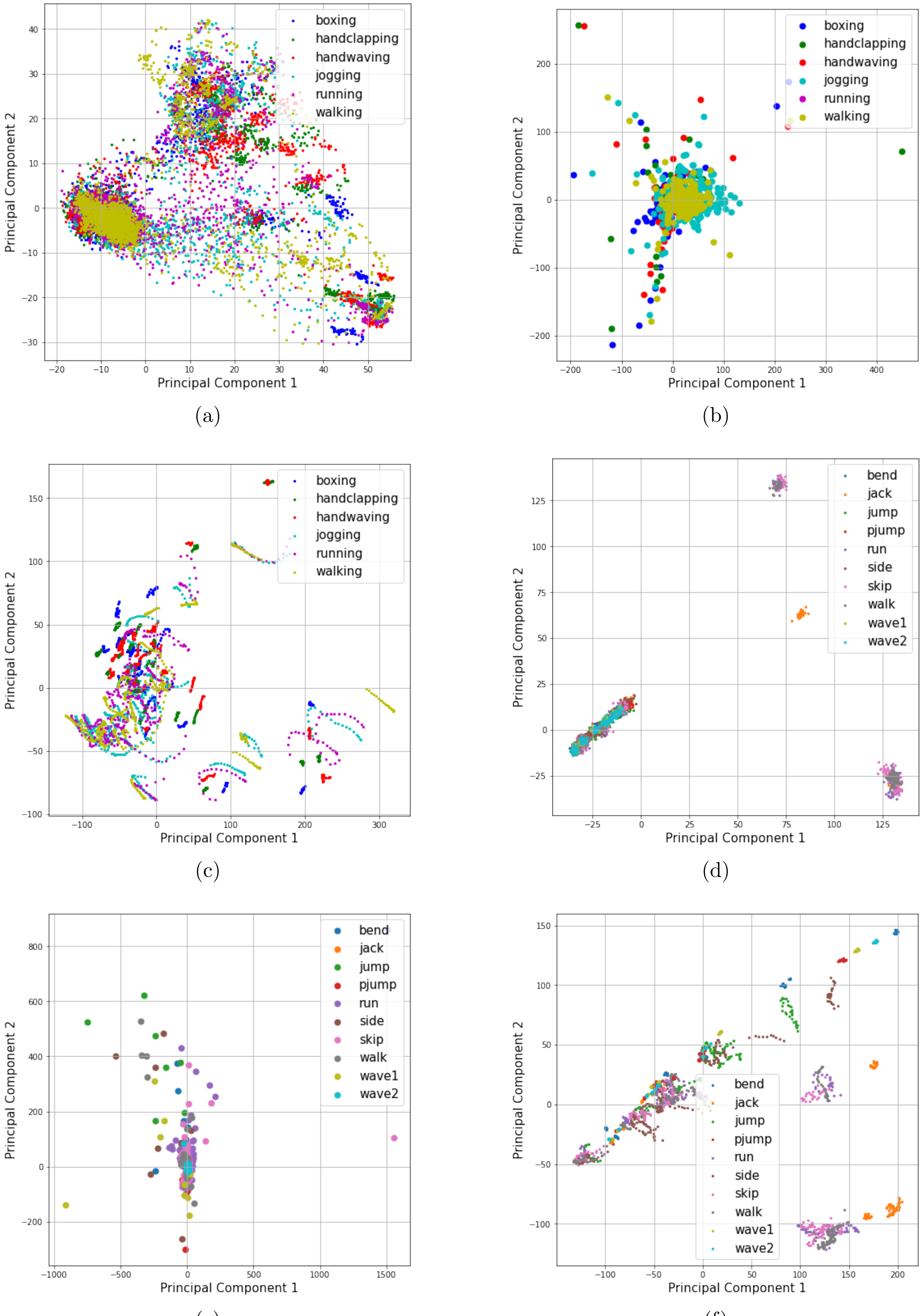
Table 4.2: Summary of evaluation results for the KTH datasets [%].

both datasets. The combination of background subtraction, NILEG, and prototypical networks shows the best accuracy (99.85%) as shown the Table 4.2. Moreover, when using a larger embedded dimension M, the classification accuracies were increased.

4.1 Experimental Results of Dimensionality Reduction

In this part, we shows more detailed results of each dimensionality reduction including PCA, ISOMAP, NuMax, NILEG, and NILED. These dimensionality reduction methods can speed up the process because the dimensions of feature extraction is too high. We visualize the results of dimensionality reduction methods in each case.

First of all, we evaluate the performance of PCA algorithm. We apply PCA to reduce that high dimensional data into 2 dimensions for the KTH and Weizmann datasets. After feature extraction methods including background subtraction, HOG, and Optical flow, the dimensions of feature extraction are between 9576 and 25920 as shown in Table 2.1. We standardize the dataset's features onto unit scale which is a requirement for the optimal performance of several machine learning algorithms. We reduce the dimensionality of the datasets to several dimensions increasing by ten intervals. The results of visualization about 2-dimensional data are different varying each feature extraction as shown in Figure 4.1.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.1: Example of PCA [(a)∼(c):KTH datasets, (d)∼(f):Weizmann datasets]. (a) HOG. (b) opticalflow. (c) background. (d) HOG. (e) opticalflow. (f) background.

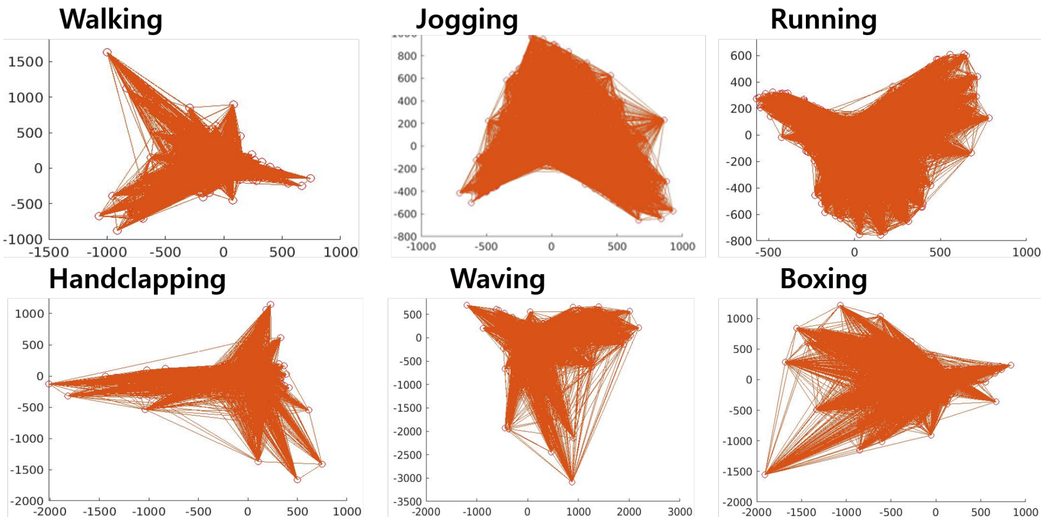
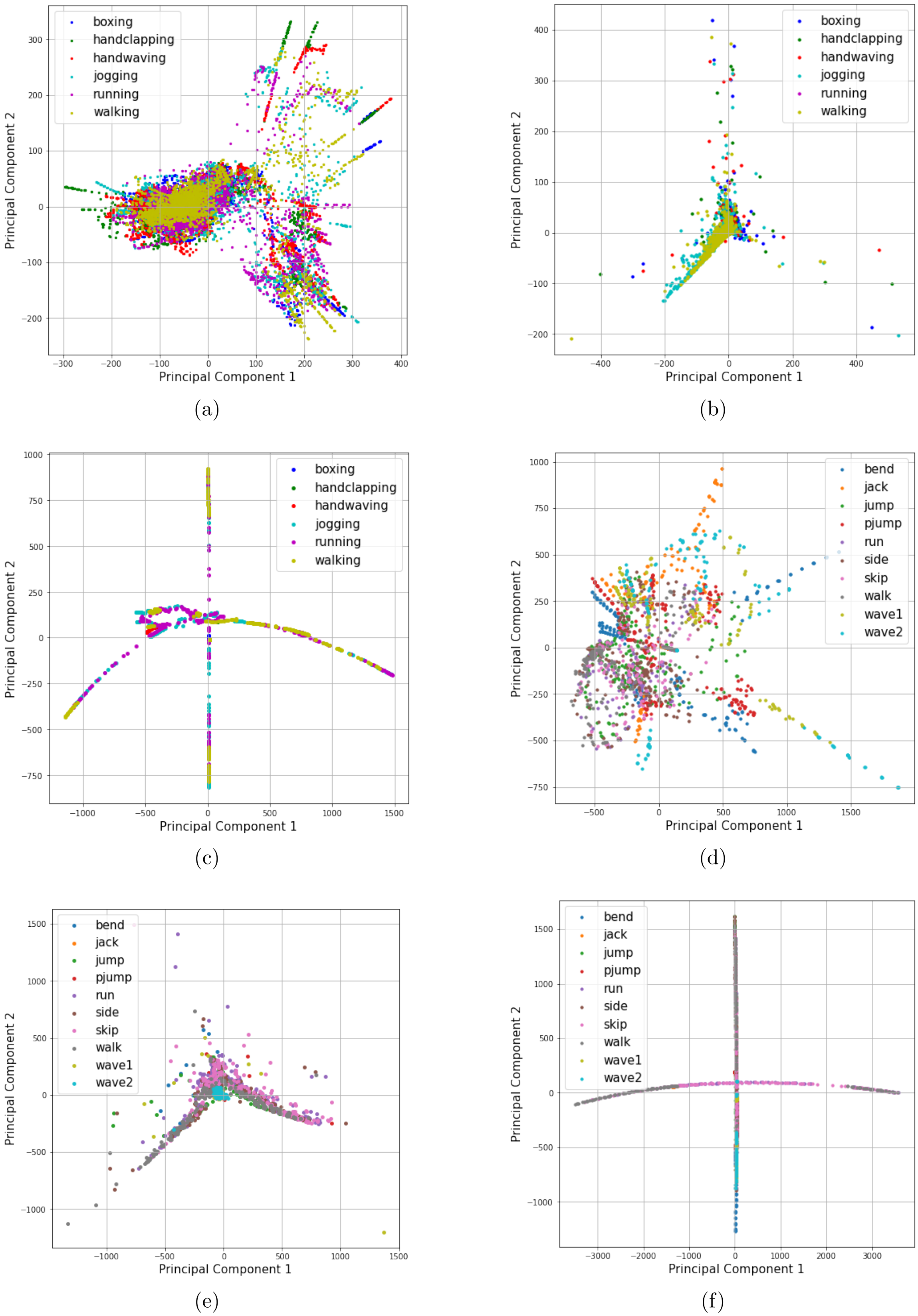


Figure 4.2: Manifolds of Activities of KTH datasets

Secondly, we use the ISOMAP to get template models of the observed human motions. ISOMAP can be used as a means of representing the actual intrinsic dimensionality of the analyzed data. We use ISOMAP to make a manifold representation of our human motion sequence [10]. The input data used by this step is a set of of silhouette images and HOG, and optical flow obtained by the preprocessing step of our method. For all methods, the local manifold similarity is based on the K-nearest neighbors. We use value of 5 as used in [10]. Each motion manifold space generated by this embedding contains two dimensions and is generated from the image without considering time information into consideration as shown in Figure 4.2. Figure 4.3 shows the results of ISOMAP for the KTH and Weizmann datasets. The ISOMAP algorithm reduces the dimensionality of the datasets to a specific dimensions for the KTH and Weizmann datasets. In constrast to HOG and optical flow, the results of background subtraction are clustered in some rows as shown in Figure 4.3(c) and 4.3(f). We apply some dimensions increasing by ten intervals as in the case of other dimensionality reduction methods.



(a)

(b)

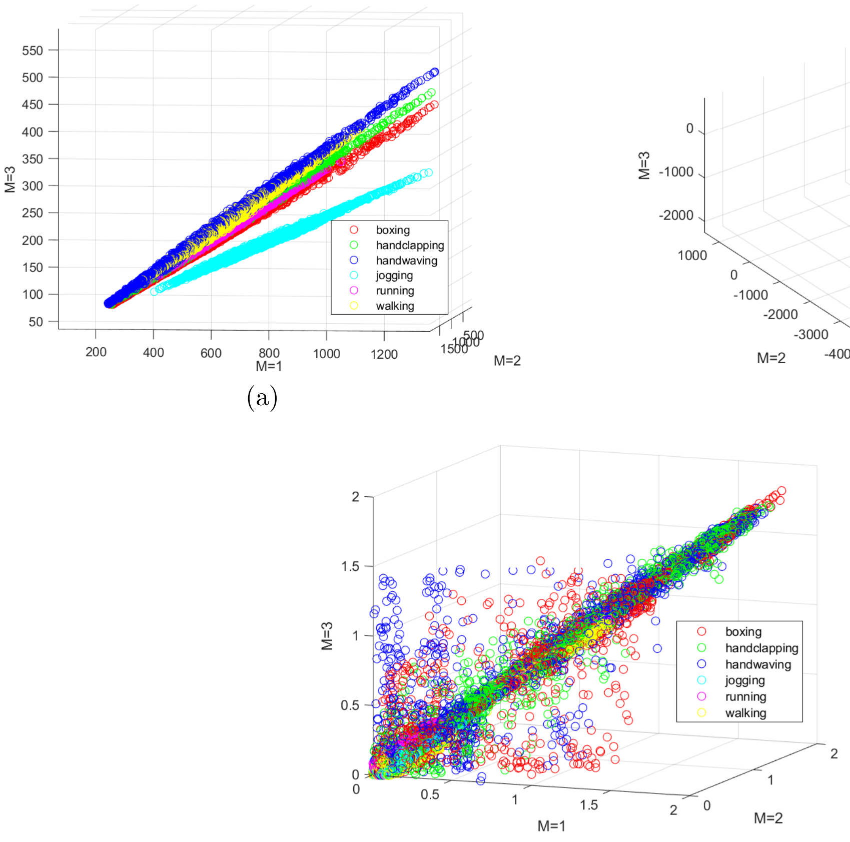
(c)

(d)

(e)

(f)

Figure 4.3: Example of ISOMAP [(a)∼(c):KTH datasets, (d)∼(f):Weizmann datasets]. (a) HOG. (b) opticalflow. (c) background. (d) HOG. (e) opticalflow. (f) background.

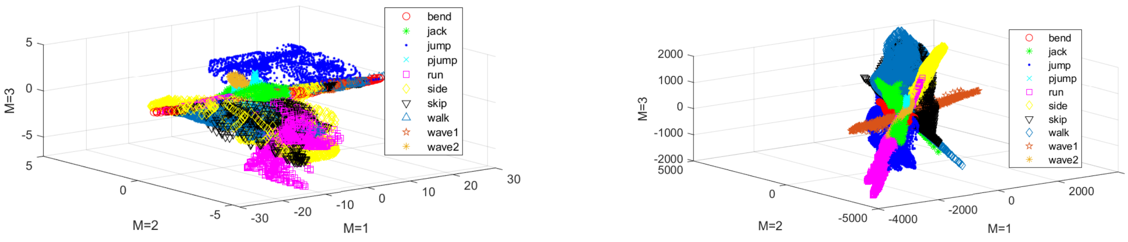


(a)

(b)

(c)

Figure 4.4: Example of NuMax for the KTH datasets. (a) HOG. (b) opticalflow. (c) background subtraction.



(a)

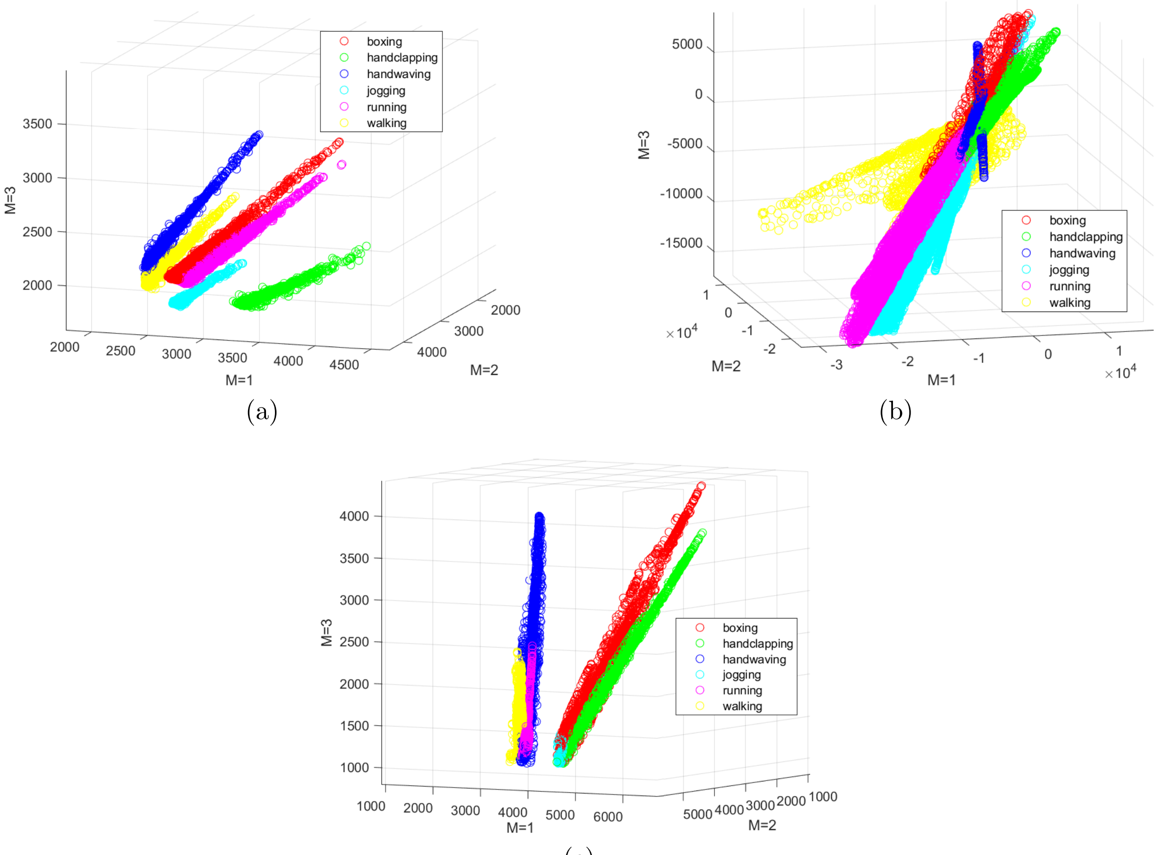
(b)

Figure 4.5: Example of NuMax on Weizmann datasets with optical flow features. (a) Before NuMax (b) After NuMax.

Figure 4.4 shows the results of NuMax for the KTH datasets. We use NuMax by varying the embedded dimensions M of the low dimensional embeddings using isometry constant δ. We manually select $\omega = 1.618$ and $\beta_1 = \beta_2 = 1$, iterations $= 1000$

like as used in [27]. Figure 4.5 shows the results of NuMax for the Weizmann datasets. The results of NuMax reduce the dimension of datasets as well as cluster each class of feature extraction. The embedded dimensions M for KTH and Weizmann datasets are manually set between 2 and 750 respectively to compare other dimensionality methods on the same conditions.

Figure 4.6 and 4.7 shows the outcome of NILEG applied to the KTH and Weizmann datasets, respectively. After background subtraction is used (Figure 4.7(a)), NILEG is applied(Figure 4.7(b)). The classes of NILEG are gathered well than other methods such as NuMax, PCA and ISOMAP. The results causes better classification accuracy of human motion datasets as mentioned Section 4.2. We demonstrate that NILEG can be applied as a dimensionality reduction for human motion videos. We first consider the motion datasets X of size 144 (height) × 180 (width) × 28 (frames). In case of background subtraction as feature extraction, we convert the $3D$ matrix ($144 \times 180 \times 28$) into $2D$ matrix (4032×180). The secant set of NILEG and NILED is made by randomly sampling pairs of dataset points from X. We test the isometry constants of NILEG and NuMax. Figure 4.8 shows the number of measurements M according to the isometry constant δ. Figure 4.8(a) shows that NILEG and NuMax achieve the desired isometry constant on the secants using a small number of measurements. We use the same secants except for normalization in NuMax and NILEG. NILEG reduces the number of measurements M more than NuMax. For example, NILEG attains a distortion of $\delta = 0.35$ with 1.5 times fewer measurements than NuMax. In Figure 4.8(b), we compare the number of measurements M according to the number of neighbors (K = 5, 15, 30). When we use the smallest K, the number of measurements M also is reduced. In other words, we can preserve the datasets using the fewer number of measurement of M. As the isometry constant δ increases, M is converged to similar values (M=2).

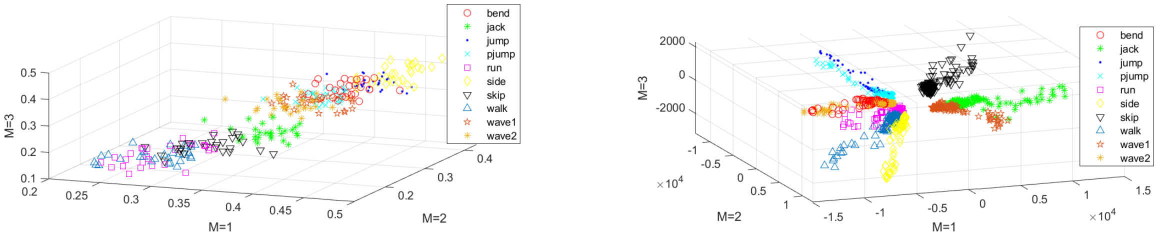


(a)

(b)

(c)

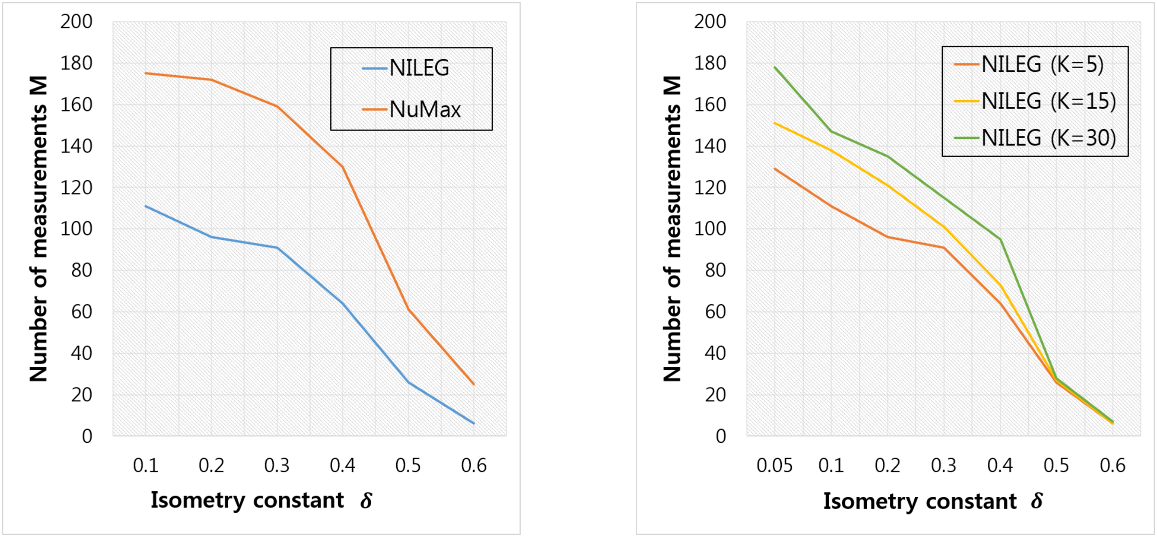
Figure 4.6: Example of LILEG for the KTH datasets. (a) HOG. (b) opticalflow. (c) background subtraction.



(a)

(b)

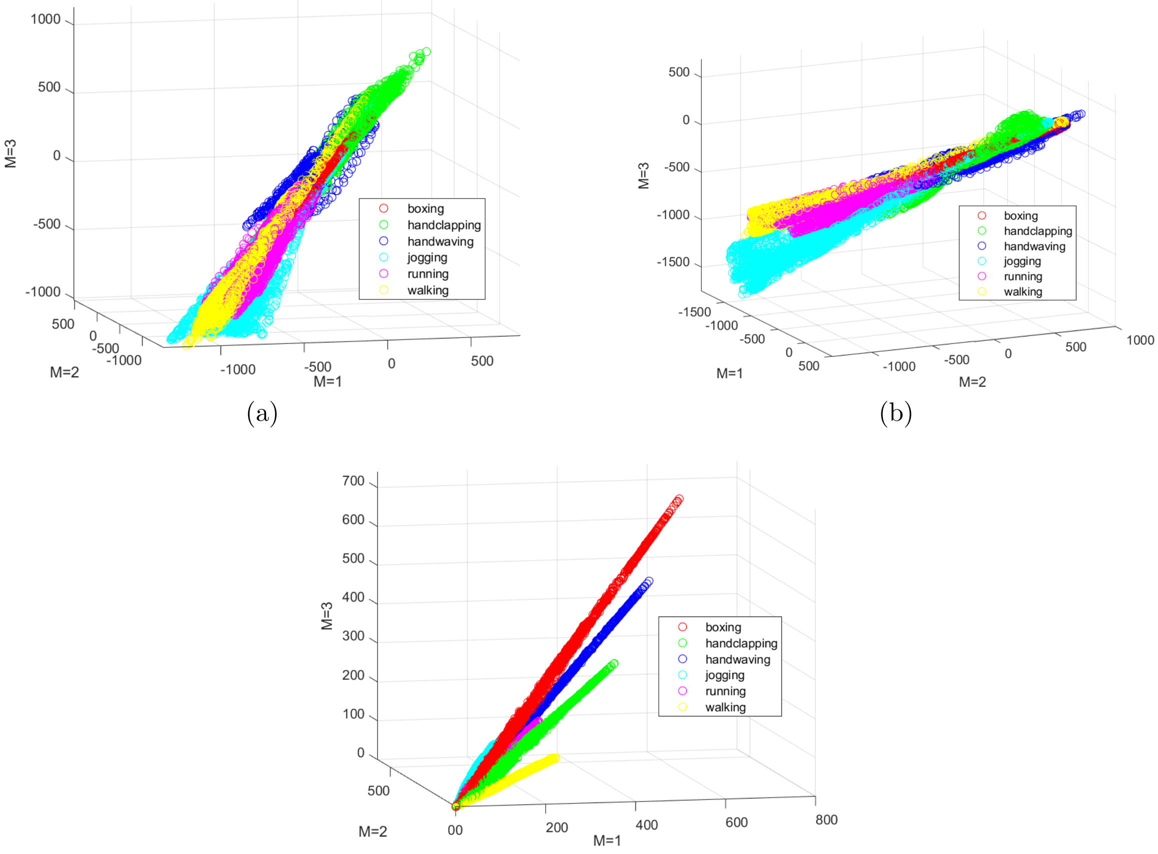
Figure 4.7: Example of NILEG on Weizmann datasets with background subtraction features. (a) Before NILEG (b) After NILEG.



(a) (b)

Figure 4.8: Isometry constant δ vs. Number of measurements M. (a) NILEG and NuMax. (b) NILEG(Nearest neighbors K = 5, 15, 30).

Figure 4.9 shows the example of NILED for the KTH datasets. We test our method on a set of human motion sequences widely used in the literature [10]. We divided the data set into training subset and testing subset. These two subsets cover all individuals performing all actions for the KTH and Weizmann datasets. The result of NILED through background subtraction shows that the datasets are gathered well than HOG and optical flow. The results cause high accuracy of classification in Section 4.2. In contrast to existing research [10], we construct secant set based on DTW distance through neighbor sequences for each activity for the comparison to NuMax and LILEG. We prove the efficiency of DTW distance as well as geodesic distance in HMR. DTW can be successfully used for matching motion patterns of embedded manifolds as shown Figure 4.9 and 4.10. Although previous research use DTW as a classifier [55], our paper aims at demonstrating the effectiveness of the feature extraction-DTW-FSL combinations. We use the same parameters like isometry constant δ to make holistic comparisons with NuMax and NILEG.

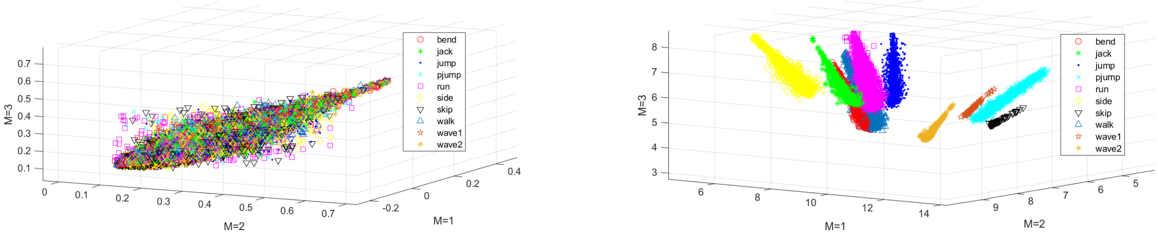


(a)

(b)

(c)

Figure 4.9: Example of NILED for the KTH datasets. (a) HOG. (b) opticalflow. (c) background subtraction.



(a)

(b)

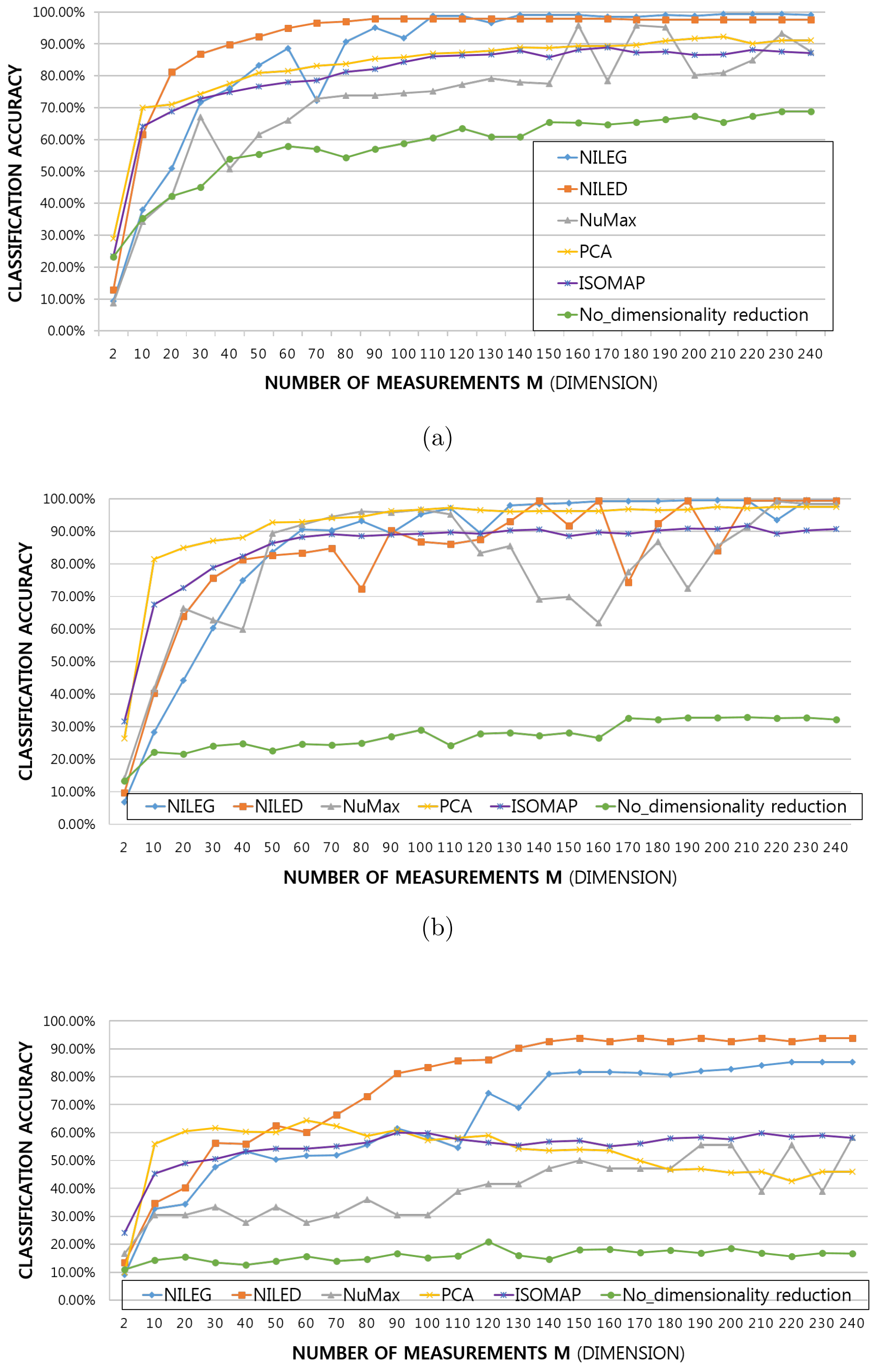
Figure 4.10: Example of NILED on Weizmann datasets with HOG features. (a) Before NILED (b) After NILED.

4.2 Classification Performance

In this section, we conduct experiments of several combinations on two datasets to testify our proposed methods for HMR. We implement existing methods to make comparisons with NILEG (geodesic distance) and NILED (DTW), including the classical method PCA and ISOMAP as dimensionality reduction methods.

Figure 4.11 shows the results of classification accuracy for various methods using the large margin networks for the Weizmann datasets. We use three different methods: space-time silhouettes, the histogram of oriented gradients and human optical flow for feature extraction as mentioned in Section 2.1. We then apply various dimensionality reduction techniques including NILEG and NILED. We also compare to the performance when no dimensionality reduction is performed. In Figure 4.11, as the number of measurements M increases, the accuracy tends to improve gradually.

Our experimental results generally show that NILEG obtains the highest classification accuracy. As shown in Figure 4.11(a), there are a variety of performance levels when M varies between 2 and 80. For example, NILED gets the highest accuracy between $M = 20$ and $M = 100$. In the case of $M \geq 110$, NILEG performs almost perfectly with classification accuracy of 99.42%. In Figure 4.11(b), we show results when HOG is used for feature extraction and several dimensionality reduction methods as M varies. NILEG outperforms other methods once the number of measurements M is sufficiently large. The best classification accuracy is 99.47% through HOG, NILEG, and large margin networks. When we use optical flow for feature extraction as shown in Figure 4.11(c), the classification accuracy is mostly lower than for competing feature extraction methods. The best classification result is 92.29% through NILED. In case of optical flow (Figure 4.11(c)), we use any of the pair (V_x, V_y) or (Orientation, Magnitude) as a reference to check if an object is moving. We cluster the points which have same (V_x, V_y) in a region that might represent an object because for a single object its velocity for every point remain same.



(a)

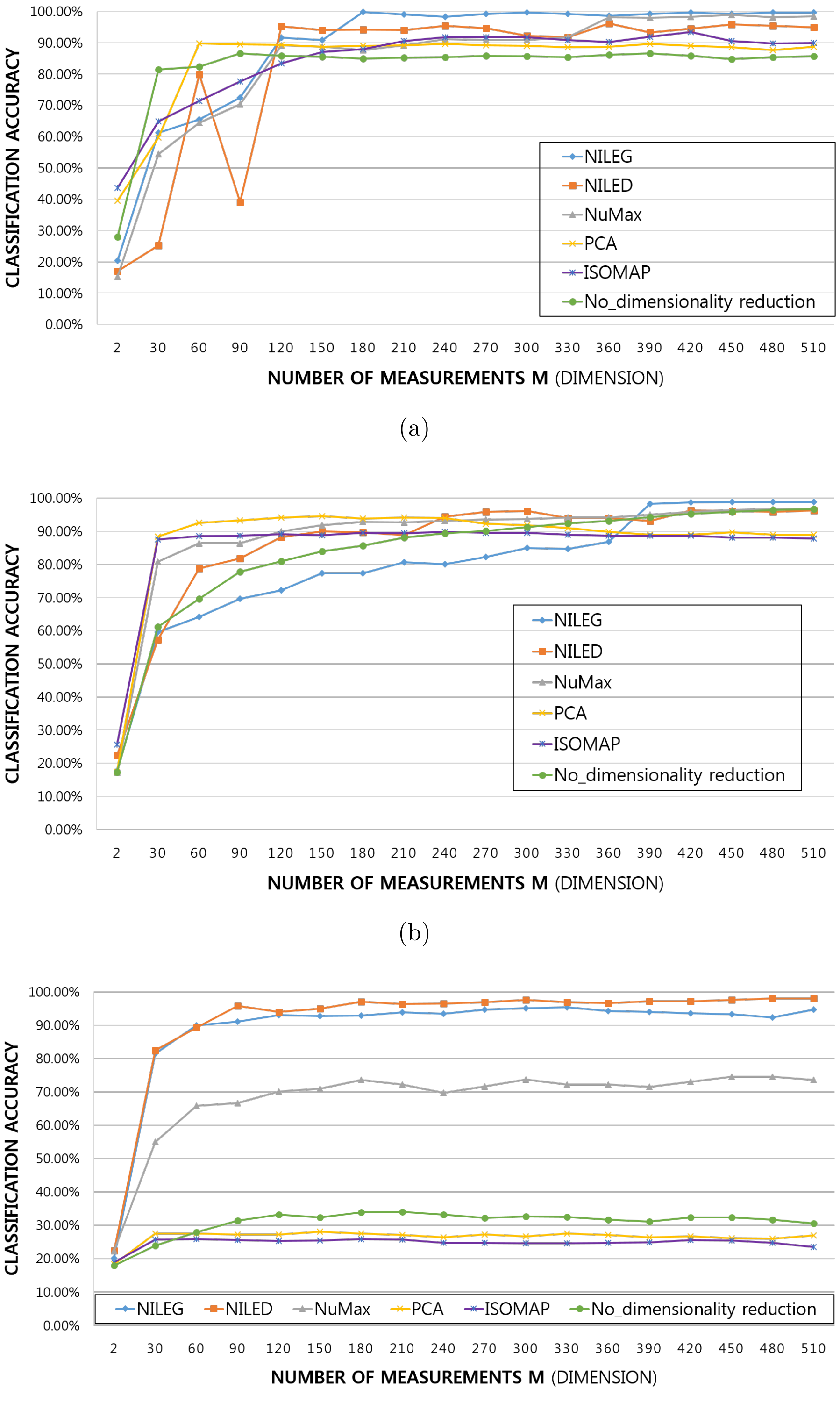
(b)

(c)

Figure 4.11: Classification Accuracy for Large Margin Networks for the Weizmann datasets using different feature extraction and dimensionality reduction algorithms. (a) Background Subtraction. (b) HOG. (c) Optical flow.

We also use the different datasets to escape data dependence and prove the performance of various methods including NILEG and NILED. Figure 4.12 shows the results of the classification errors for various methods for the KTH datasets. When we use background subtraction as feature extraction to KTH datasets as shown in Figure 4.12(a), the classification performance of NuMax and PCA is higher than NILEG for some values of M. (For example, the classification accuracy of NILEG : 65.52%, NILED : 79.89%, NuMax : 64.37%, PCA : 89.83% for $M = 60$). However, NILEG generally has the best classification accuracy when being applied background subtraction when the embedded dimension $M > 180$. Next, the accuracies of NILED are higher than NuMax, PCA, and ISOMAP.

NILEG also has the best classification accuracy when being applied HOG when the embedded dimension $M > 390$. The best classification accuracy is 98.25% through NILEG, HOG and large margin networks when the embedded dimension $M = 390$. The results show that NILEG performs the best classification among the tested embedding methods. As shown in Figure 4.12(c), we get the classification accuracy 97.95% through optical flow, NILED and large margin networks in $M = 480$. In case of optical flow as feature extraction, NILED has the best classification performance among all methods for all embedding dimensions in shown as Figure 4.12(c). These empirical results show that the secant set based on geodesic distance and DTW preserve human motion metrics better rather than other dimensionality reduction methods. This is because these methods better preserves the properties of human motion datasets captured by geodesic and DTW distance. Our experimental results generally show that NILED obtains the highest classification accuracy when being applied optical flow for two datasets as shown in Figure 4.11(c) and 4.12(c).

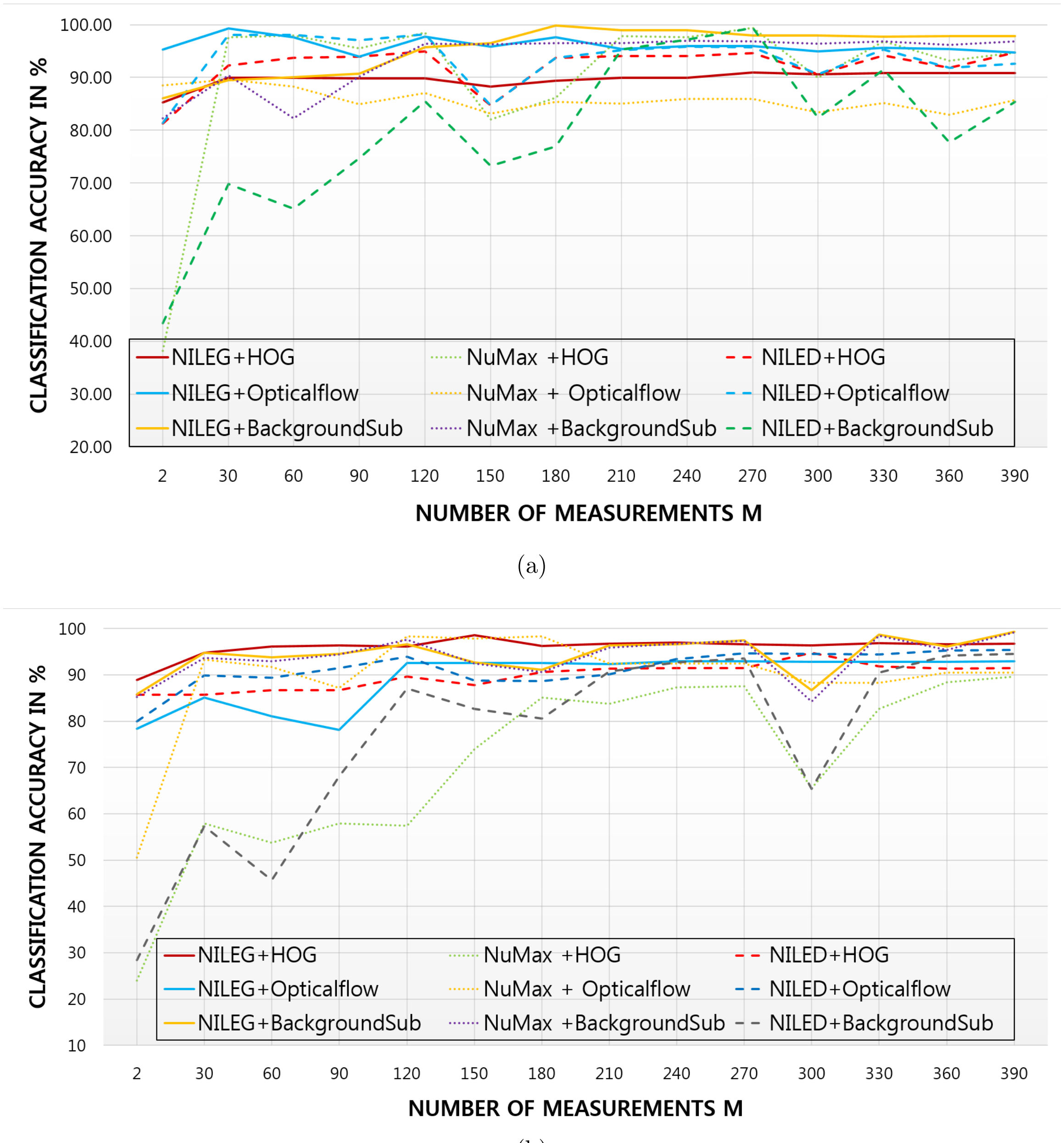


(a)

(b)

(c)

Figure 4.12: Classification error for Large Margin Networks for the KTH datasets using different feature extraction and dimensionality reduction algorithms. (a) Background Subtraction. (b) HOG. (c) Optical flow.

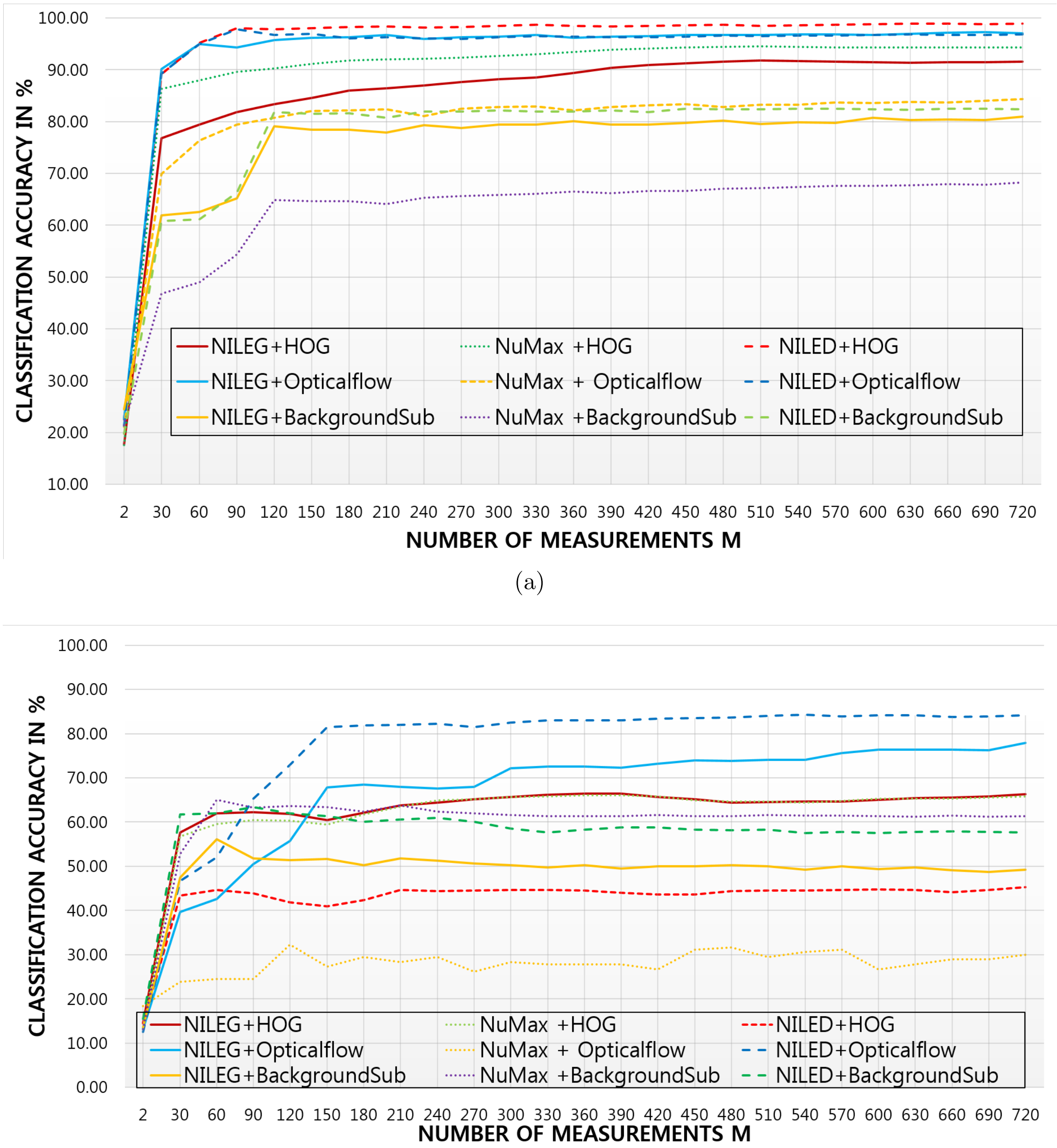


(a)

(b)

Figure 4.13: Evaluation of prototypical networks. (a) KTH datasets. (b) Weizmann datasets.

Figure 4.13 describes experimental results from nine methods using prototypical networks. For the KTH datasets, we get the best performance (99.85%) using NILEG for dimensionality reduction and background subtraction for feature extraction when using the embedded dimension $M = 180$ as shown in Figure 4.13(a). The best performing methods in terms of classification accuracy are different for the Weizmann datasets and the KTH datasets. The combination of NILEG and HOG has the highest classification accuracy (98.62%) in the case $M = 150$ for the Weizmann datasets. The fluctuation of classification accuracy of NILED + background subtraction is larger than other methods according to M. Figure 4.14 shows the performance of matching networks using NILEG, NuMax and NILED. The evaluation was done using a 5-fold cross-validation technique for an out-of-sample test. The combination of NILED for dimensionality reduction and HOG for feature extraction have the best classification accuracy among different five combinations to whole embedded dimension M in KTH datasets as shown in Figure 4.14(a). When having a larger embedded dimension M, NILED+HOG mostly outperforms other combinations including NILEG + HOG, NuMax + HOG, and NuMax + Background subtraction. The combination of NuMax + Background subtraction shows the worst performance for the KTH datasets as shown in Figure 4.14(a). For the Weizmann datasets, the combination of NILED and optical flow outperforms other methods when the embedding dimensionality $M \geq 150$ as shown the Figure 4.14(b). The average classification accuracies of the matching networks are lower than large margin networks and prototypical networks as discussed in section 3.2. The best classification accuracy on the Weizmann dataset is 84.24% for $M = 540$. Generally, we get higher accuracy for a larger dimension M with lower isometry constant δ. As seen from the above, our proposed approach outperforms those existing in the literature. Therefore, NILEG and NILED can be a good choice for dimensionality reduction in human motion datasets.



(a)

(b)

Figure 4.14: Evaluation of matching networks. (a) KTH datasets. (b) Weizmann datasets.

CHAPTER 5

CONCLUSIONS

Weizmann dataset		KTH dataset	
Methods	Accuracy [%]	Methods	Accuracy [%]
Fathi et al. [23]	90.00	Schuldt et al. [48]	71.70
Ali et al. [3]	94.75	Dollar et al. [21]	81.20
Bregonzio et al. [14]	96.66	Niebles et al. [42]	83.30
Seo. et al. [49]	97.50	Jhuang et al. [31]	91.70
Wang et al. [63]	96.70	Ji et al. [33]	90.20
Arac et al. [1]	97.77	Schindler et al. [47]	92.70
Fadwa et al. [2]	97.02	Arac et al. [1]	95.36
Our best method	**99.47**	**Our best method**	**99.85**

Table 5.1: The results of Human motion recognition [%].

In this research, we propose novel metric-based few shot learning via linear embeddings for human motion recognition. The main contributions of this proposed approach are two-fold. First, we propose the first application for combinations of feature extraction, dimensionality reduction, and metric-based few-shot learning to improve the classification performance for human motion datasets. Contrary to the usual methods of classifiers such as support vector machine [16, 23, 50], we apply metric learning-based classifiers such as large margin networks to increase the classification accuracy for human motion recognition. Metric learning based on deep neural networks provides good performance, however, they are often computationally inefficient and the network training requires significant amounts of training data and computation. To compare the performance of neural networks with our methods, we used a video classification model by combining a pre-trained image classification

Division	*NILEG*	*NILED*	*CNN [36]*	*PCA*	*ISOMAP*
Computation times (Minuites)	24.21	15.64	49.47	**5.01**	10.36
Classification Accuracy (%)	**99.47**	99.31	88.89	88.11	90.48

Table 5.2: Computation times and classification accuracy of five methods.

and an LSTM network provided by MATLAB [40]. GoogLeNet Network [56] is used as pre-trained deep learning toolbox. As shown in Table 5.2, the computational times of convolutional neural networks (CNN) were 49.47 minutes for computation of classification accuracy through background subtraction to training with classification accuracy 88.89%. The computational times of NILEG and ISOMAP took less than CNN, the classification accuracy of NILEG and ISOMAP was higher than CNN. Our proposal makes a simple and general framework for human motion recognition.

Second, the highest classification accuracy can be achieved even when a few samples are available for query class based on few-shot learning, in particular, in the case of prototypical networks. Although our methods do not use state-of-the-art techniques such as Deep Neural Networks (DNN) or Convolutional Neural Networks (CNN), we get better results for human motion recognition as shown in Table 5.1. We assess the capability of our combination with linear embedding and metric-based few-shot learning to provide features that allow for reliable recognition of the human motion from small datasets. Unlike related works [46], we use NILEG using a secant set based on geodesic distance as linear embeddings. We compare several embeddings techniques to discover the best results. NILEG preserves the properties of human motion datasets and designs efficient and scalable algorithms for embedding. This motivates specific computational challenges and solutions. We use the respective merits of linear embeddings and metric-based few-shot learning. Our method might widen an area from images classification to human motion analysis using small datasets. Although existing works in the field of few-shot learning are related to image classification, this work can be used to effectively study human motion recognition, prediction, and other applications. We obtain high performance for human motion recognition through the best combinations.

We discuss some future aspects in the field of human vision. We will study human motion prediction beyond the classification. On the contrary to motion classification, human motion prediction focuses on future scenarios. We may combine various methods such as metric learning as well as motion trajectory for short-term and long-term prediction. Besides, it is worth trying to study from multi-modal data including videos as well as audio and text, etc. The analytical methods of complex action datasets can be improved the performance of human motion classification and prediction [37]. Finally, learning human motions without labels for more efficient solutions can be studied in computer vision.

BIBLIOGRAPHY

[1] Acar, Esra, Senst, Tobias, Kuhn, Alexander, Keller, Ivo, Theisel, Holger, Albayrak, Sahin, and Sikora, Thomas. Human action recognition using lagrangian descriptors. *2012 IEEE 14th International Workshop on Multimedia Signal Processing (MMSP)* (2012), 360–365.

[2] Al-Azzo, Fadwa, Bao, Chunbo, Taqi, Arwa Mohammed, Milanova, Mariofanna G., and Ghassan, Nabeel. Human actions recognition based on 3d deep neural network. *2017 Annual Conference on New Trends in Information Communications Technology Applications (NTICT)* (2017), 240–246.

[3] Ali, Saad, and Shah, Mubarak. Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32* (2010), 288–303.

[4] Alzughaibi, Arwa, Hakami, Hanadi, and Chaczko, Zenon. Review of human motion detection based on background subtraction techniques. *International Journal of Computer Applications 122* (2015), 1–5.

[5] Bachmann, C. M., Ainsworth, T. L., Fusina, R. A., Topping, R., and Gates, T. Manifold coordinate representations of hyperspectral imagery: Improvements in algorithm performance and computational efficiency. In *2010 IEEE International Geoscience and Remote Sensing Symposium* (2010), pp. 4244–4247.

[6] Baraniuk, Richard G., Davenport, Mark A., DeVore, Ronald A., and Wakin, Michael B. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation 28* (2008), 253–263.

[7] Bellet, Aurélien, Habrard, Amaury, and Sebban, Marc. A survey on metric learning for feature vectors and structured data. *ArXiv abs/1306.6709* (2013).

[8] Bengio, Yoshua, Paiement, Jean-François, Vincent, Pascal, Delalleau, Olivier, Roux, Nicolas Le, and Ouimet, Marie. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In *NIPS* (2003).

[9] Binder, Marc D., Hirokawa, Nobutaka, and Windhorst, Uwe, Eds. *Aperture Problem*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 159–159.

[10] Blackburn, Jaron, and Ribeiro, Eraldo. Human motion recognition using isomap and dynamic time warping. In *Workshop on Human Motion* (2007).

[11] Blank, Moshe, Gorelick, Lena, Shechtman, Eli, Irani, Michal, and Basri, Ronen. Actions as space-time shapes. *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 2* (2005), 1395–1402 Vol. 2.

[12] Blank, Moshe, Gorelick, Lena, Shechtman, Eli, Irani, Michal, and Basri, Ronen. Actions as space-time shapes. In *The Tenth IEEE International Conference on Computer Vision (ICCV'05)* (2005), pp. 1395–1402.

[13] Boley, Daniel. Linear convergence of admm on a model problem.

[14] Bregonzio, Matteo, Xiang, Tao, and Gong, Shaogang. Fusing appearance and distribution information of interest points for action recognition. *Pattern Recognit. 45* (2012), 1220–1234.

[15] Broomhead, David S., and Kirby, Michael J. The whitney reduction network: A method for computing autoassociative graphs. *Neural Computation 13* (2001), 2595–2616.

[16] Cao, Dongwei, Masoud, Osama, Boley, Daniel, and Papanikolopoulos, Nikolaos. Human motion recognition using support vector machines. *Comput. Vis. Image Underst. 113* (2009), 1064–1075.

[17] Chang, Jia-Ren, and Chen, Yong-Sheng. Pyramid stereo matching network. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), 5410–5418.

[18] Coskun, Huseyin, Tan, David Joseph, Conjeti, Sailesh, Navab, Nassir, and Tombari, Federico. Human motion analysis with deep metric learning. In *ECCV* (2018).

[19] Dai, Y., and Li, H. Rank minimization or nuclear-norm minimization: Are we solving the right problem? In *2014 International Conference on Digital Image Computing: Techniques and Applications (DICTA)* (2014), pp. 1–8.

[20] Dalal, Navneet, and Triggs, Bill. Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) 1* (2005), 886–893 vol. 1.

[21] Dollár, Piotr, Rabaud, Vincent, Cottrell, Garrison W., and Belongie, Serge J. Behavior recognition via sparse spatio-temporal features. *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance* (2005), 65–72.

[22] Farnebäck, Gunnar. Two-frame motion estimation based on polynomial expansion. In *SCIA* (2003).

[23] Fathi, Alireza, and Mori, Greg. Action recognition by learning mid-level motion features. *2008 IEEE Conference on Computer Vision and Pattern Recognition* (2008), 1–8.

[24] Fazel, Maryam, Hindi, H., and Boyd, S. A rank minimization heuristic with application to minimum order system approximation. *Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148) 6* (2001), 4734–4739 vol.6.

[25] Fei, Nanyi, Lu, Zhiwu, Gao, Yizhao, Tian, Jia, Xiang, Tao, and Wen, Ji-Rong. Meta-learning across meta-tasks for few-shot learning, 02 2020.

[26] Gorelick, Lena, Blank, Moshe, Shechtman, Eli, Irani, Michal, and Basri, Ronen. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence 29*, 12 (December 2007), 2247–2253.

[27] Hegde, Chinmay, Sankaranarayanan, Aswin C., Yin, Wotao, and Baraniuk, Richard G. Numax: A convex approach for learning near-isometric linear embeddings. *IEEE Transactions on Signal Processing 63* (2015), 6109–6121.

[28] Hochreiter, Sepp, Younger, A. Steven, and Conwell, Peter R. Learning to learn using gradient descent. In *ICANN* (2001).

[29] Huttunen, Heikki. Sahar husseini a survey of optical flow techniques for object tracking.

[30] Ikizler, Nazli, and Sahin, Pinar Duygulu. Human action recognition using distribution of oriented rectangular patches. In *Workshop on Human Motion* (2007).

[31] Jhuang, Hueihan, Serre, Thomas, Wolf, Lior, and Poggio, Tomaso A. A biologically inspired system for action recognition. *2007 IEEE 11th International Conference on Computer Vision* (2007), 1–8.

[32] Ji, Shuiwang, Xu, Wei, Yang, Ming, and Yu, Kai. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35* (2010), 221–231.

[33] JiShuiwang, Xuwei, Yang-ming, and Yu-kai. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2013).

[34] Kalouris, Gerasimos, Zacharaki, Evangelia I., and Megalooikonomou, Vasileios. Improving cnn-based activity recognition by data augmentation and transfer learning. *2019 IEEE 17th International Conference on Industrial Informatics (INDIN) 1* (2019), 1387–1394.

[35] Khan, M. M. R., Arif, R. B., Siddique, M. A. B., and Oishe, M. R. Study and observation of the variation of accuracies of knn, svm, lmnn, enn algorithms on eleven different datasets from uci machine learning repository. In *2018 4th International Conference on Electrical Engineering and Information Communication Technology (iCEEiCT)* (2018), pp. 124–129.

[36] Kimmel, Ron, and Sethian, James A. Computing geodesic paths on manifolds. *Proceedings of the National Academy of Sciences of the United States of America 95 15* (1998), 8431–5.

[37] Kong, Yu, and Fu, Yun. Human action recognition and prediction: A survey. *ArXiv abs/1806.11230* (2018).

[38] Köppen, Mario. The curse of dimensionality.

[39] Liang, Yu-Ming, Shih, Sheng-Wen, and Shih, Arthur Chun-Chieh. Human action segmentation and classification based on the isomap algorithm. *Multimedia Tools and Applications 62* (2011), 561–580.

[40] The Mathworks, Inc. *MATLAB version 9.3.0.713579 (R2017b).* Natick, Massachusetts, 2017.

[41] Mensink, Thomas, Verbeek, Jakob J., Perronnin, Florent, and Csurka, Gabriela. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35* (2013), 2624–2637.

[42] Niebles, Juan Carlos, Wang, Hongcheng, and Li, Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision 79* (2007), 299–318.

[43] Ranjan, Anurag, Romero, Javier, and Black, Michael J. Learning human optical flow. In *BMVC* (2018).

[44] Ravichandiran, S. *Hands-On Meta Learning with Python: Meta learning using one-shot learning, MAML, Reptile, and Meta-SGD with TensorFlow.* Packt Publishing, 2018.

[45] Recht, Benjamin, Fazel, Maryam, and Parrilo, Pablo A. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review 52* (2010), 471–501.

[46] Roweis, Sam T., and Saul, Lawrence K. Nonlinear dimensionality reduction by locally linear embedding. *Science 290 5500* (2000), 2323–6.

[47] Schindler, Konrad, and Gool, Luc Van. Action snippets: How many frames does human action recognition require? *2008 IEEE Conference on Computer Vision and Pattern Recognition* (2008), 1–8.

[48] Schüldt, Christian, Laptev, Ivan, and Caputo, Barbara. Recognizing human actions: a local svm approach. *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. 3* (2004), 32–36 Vol.3.

[49] Seo, Hae Jong, and Milanfar, Peyman. Action recognition from one example. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33* (2011), 867–882.

[50] Sidenbladh, H. Detecting human motion with support vector machines. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* (2004), vol. 2, pp. 188–191 Vol.2.

[51] Snell, Jake, Swersky, Kevin, and Zemel, Richard S. Prototypical networks for few-shot learning. *ArXiv abs/1703.05175* (2017).

[52] society IEEE, Computer. Conference on Computer Vision and Pattern Recognition. `http://cvpr2019.thecvf.com/`, 2019. [Online; accessed 19-April-2020].

[53] Sulaiman, Sarina, Hussain, Abrar, Tahir, N. M., Muad, A. M., and Mustafa, M. M. Scene analysis for human silhoutte extraction. *2006 4th Student Conference on Research and Development* (2006), 124–126.

[54] Sun, Weiwei, Yang, Gang, Du, Bo, Zhang, Lefei, and Zhang, Liangpei. A sparse and low-rank near-isometric linear embedding method for feature extraction in hyperspectral imagery classification. *IEEE Transactions on Geoscience and Remote Sensing 55* (2017), 4032–4046.

[55] Switonski, Adam, Josinski, Henryk, and Wojciechowski, Konrad. Dynamic time warping in classification and selection of motion capture data. *Multidimensional Systems and Signal Processing 30* (2019), 1437–1468.

[56] Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)* (2015).

[57] Tamgade, Sukeshni N., and Bora, Vibha R. Motion vector estimation of video image by pyramidal implementation of lucas kanade optical flow. *2009 Second International Conference on Emerging Trends in Engineering Technology* (2009), 914–917.

[58] Tenenbaum, Joshua B. The isomap algorithm and topological stability. *Science 295* (2002).

[59] Uijlings, Jasper R. R., Duta, I. C., Sangineto, Enver, and Sebe, Nicu. Video classification with densely extracted hog/hof/mbh features: an evaluation of the accuracy/computational efficiency trade-off. *International Journal of Multimedia Information Retrieval 4* (2014), 33–44.

[60] Vilalta, Ricardo, Giraud-Carrier, Christophe G., and Brazdil, Pavel. Metalearning - concepts and techniques. In *Data Mining and Knowledge Discovery Handbook* (2010).

[61] Vinyals, Oriol, Blundell, Charles, Lillicrap, Timothy P., Kavukcuoglu, Koray, and Wierstra, Daan. Matching networks for one shot learning. In *NIPS* (2016).

[62] Wang, Heng, Kläser, Alexander, Schmid, Cordelia, and Liu, Cheng-Lin. Action recognition by dense trajectories. *CVPR 2011* (2011), 3169–3176.

[63] Wang, Heng, Ullah, Muhammad Muneeb, Kläser, Alexander, Laptev, Ivan, and Schmid, Cordelia. Evaluation of local spatio-temporal features for action recognition. In *BMVC* (2009).

[64] Wang, Limin, Xiong, Yuanjun, Wang, Zhe, Qiao, Yu, Lin, Dahua, Tang, Xiaoou, and Gool, Luc Van. Temporal segment networks: Towards good practices for deep action recognition. *ArXiv abs/1608.00859* (2016).

[65] Wang, Yaqing, Yao, Quanming, Kwok, James T., and Ni, Lionel M. Generalizing from a few examples: A survey on few-shot learning.

[66] Weinberger, Kilian Q., and Saul, Lawrence K. Distance metric learning for large margin nearest neighbor classification. In *NIPS* (2005).

[67] Wikipedia. Dynamic time warping — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Dynamic\%20time\%20warping&oldid=952656880`, 2020. [Online; accessed 10-May-2020].

[68] Zhang, Yixia, Fang, Min, and Wang, Nian. Channel-spatial attention network for fewshot classification. *PLoS ONE 14* (2019).